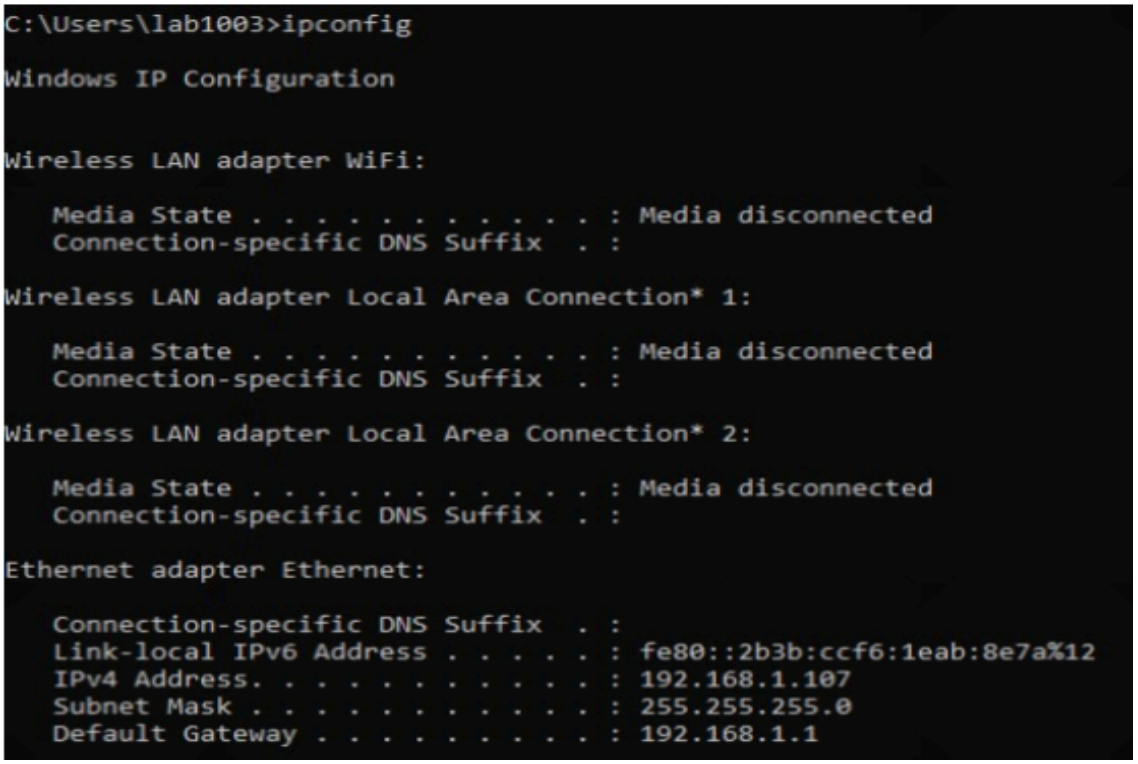
# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

**AIM** : To understand the basic commands of windows and Unix/Linux operating systems. **BASED ON LO1**

1) ***ipconfig*** (windows) :

**IPCONFIG** stands for Internet Protocol Configuration. This is a command-line application which displays all the current TCP/IP (Transmission Control Protocol/Internet Protocol) network configuration, refreshes the DHCP (Dynamic Host Configuration Protocol) and DNS (Domain Name Server). It also displays IP address, subnet mask, and default gateway for all adapters. It is available for Microsoft Windows.

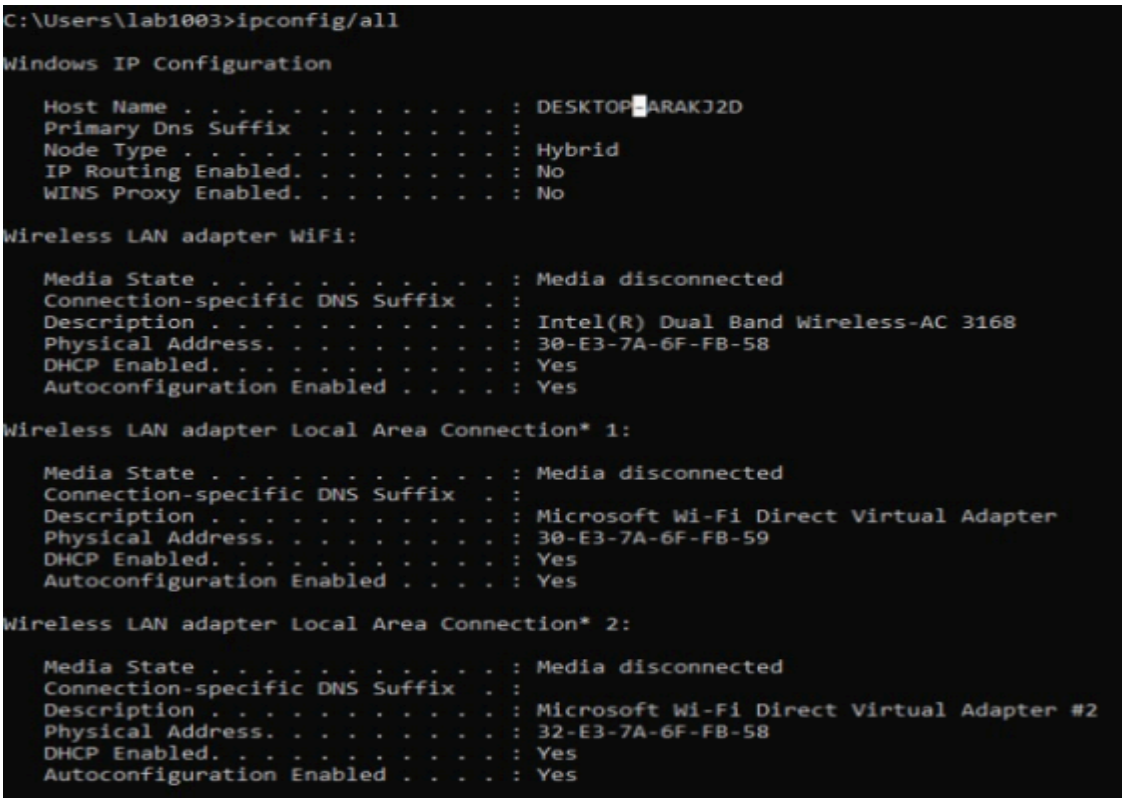
Characteristics
IPCONFIG provides settings for the IPv4 IP layer for TCP/IP.
The maximum length for the connection request queue is specified using IPCONFIG.
All the updation in the IP layer of TCP/IP is done using IPCONFIG.
IPCONFIG is used to do checksum processing for IPv4 packets

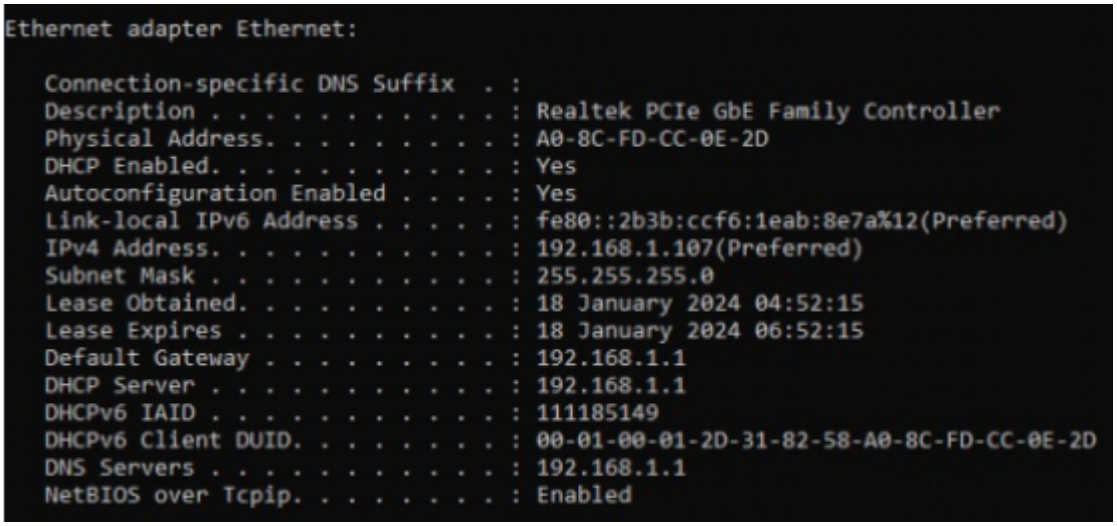


# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

2) ***ipconfig/all*** (windows) :

The **ipconfig/all** command shows all the information about your network adapter. It helps configure your connection by providing important details about the physical address, DHCP and DNS servers, the subnet mask, and default gateway, among other things. You can also see your IPv4 address and IPv6 address with this command, assuming you have both an IPv4 and IPv6 address.



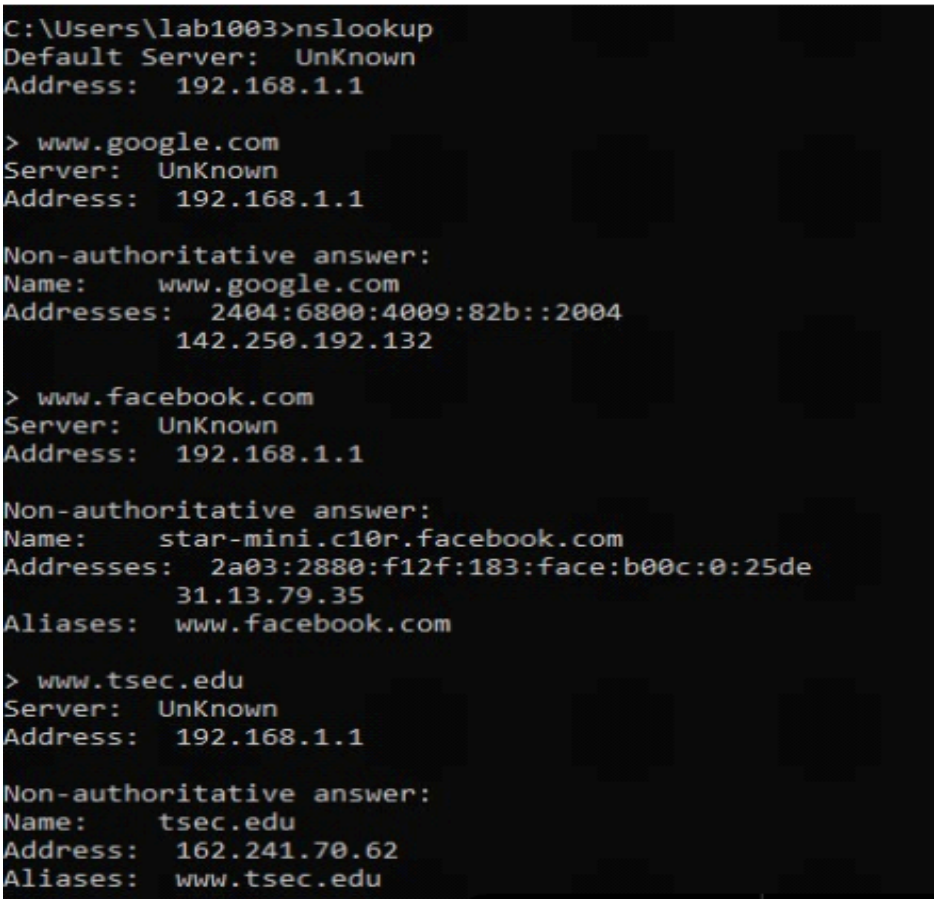



# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

3) **nslookup** (windows) :
*Purpose:* Queries internet domain name servers interactively.
*Syntax:* **nslookup** [ - option ] [ name | - ] [ server ]

*Description:*

The **nslookup** command queries internet domain name servers in two modes. Interactive mode allows you to query name servers for information about various hosts and domains, or to print a list of the hosts in a domain. In noninteractive mode,the names and requested information are printed for a specified host or domain.
The **nslookup** command enters interactive mode when no arguments are given, or when the first argument is a - (minus sign) and the second argument is the host name or internet address of a name server. When no arguments are given, the command queries the default name server. The **nslookup** command enters non-interactive mode when you give the name or internet address of the host to be looked up as the first argument. The optional second argument specifies the host name or address of a name server. You can specify options on the command line if they precede the arguments and are prefixed with a hyphen



# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

4) **ping** (windows):
The full form of PING is the **Packet InterNet Groper.** It is a computer network management system software or utility software used to test the network communication between two devices. The classification of such systems considered may include any commonly used personal computer, switch, server, gateway or router.Ping is a simple computer network software utility used to test and verify the reachability of a host on an Internet Protocol (IP) network.It works by sending packets from the source to the target host which if it is accessible through the network, then sends packets back.

Syntax : ping [/t] [/a] [/n <count>] [/l <size>] [/f] [/I <TTL>] [/v <TOS>] [/r <count>] [/s <count>] [{/j <hostlist> | /k <hostlist>}] [/w <timeout>] [/R] [/S <Srcaddr>] [/4] [/6] <targetname>

Parameters:

| | |
|---|---|
| /t | Specifies ping continue sending echo Request messages to the destination until interrupted. To interrupt and display statistics, press CTRL+ENTER. To interrupt and quit this command, press CTRL+C. |
| /a | Specifies reverse name resolution be performed on the destination IP address. If this operation is successful, ping displays the corresponding host name. |
| /n | Specifies the number of echo Request messages be sent. The default is 4. |
| /l | Specifies the length, in bytes, of the Data field in the echo Request messages. The default is 32. The maximum size is 65,500. |

***KUNAL NEMADE***

***ROLL NO :. 89***

***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

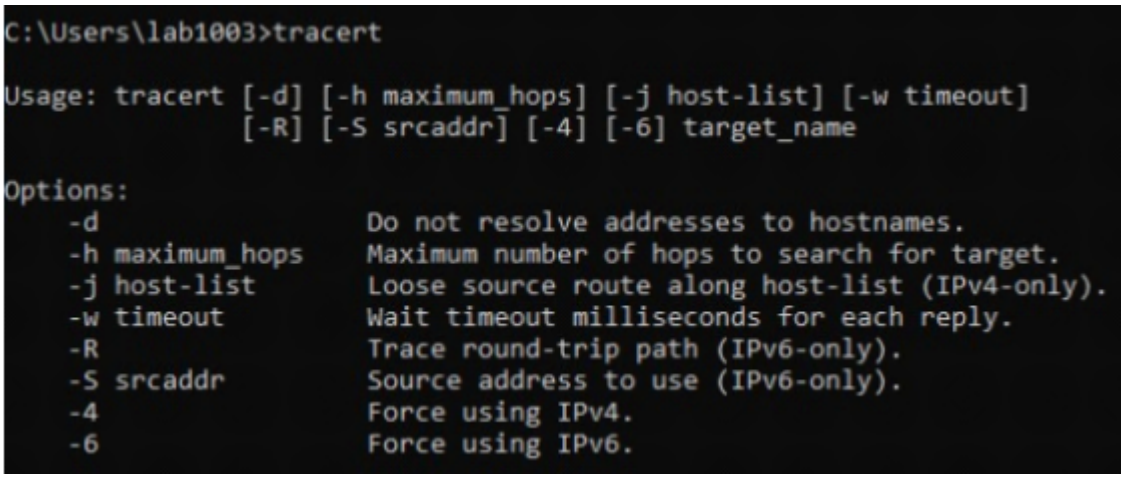
```
C:\Users\lab1003>ping

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
            [-r count] [-s count] [[-j host-list] | [-k host-list]]
            [-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
            [-4] [-6] target_name

Options:
    -t             Ping the specified host until stopped.
                   To see statistics and continue - type Control-Break;
                   To stop - type Control-C.
    -a             Resolve addresses to hostnames.
    -n count       Number of echo requests to send.
    -l size        Send buffer size.
    -f             Set Don't Fragment flag in packet (IPv4-only).
    -i TTL         Time To Live.
    -v TOS         Type Of Service (IPv4-only. This setting has been deprecated
                   and has no effect on the type of service field in the IP
                   Header).
    -r count       Record route for count hops (IPv4-only).
    -s count       Timestamp for count hops (IPv4-only).
    -j host-list   Loose source route along host-list (IPv4-only).
    -k host-list   Strict source route along host-list (IPv4-only).
    -w timeout     Timeout in milliseconds to wait for each reply.
    -R             Use routing header to test reverse route also (IPv6-only).
                   Per RFC 5095 the use of this routing header has been
                   deprecated. Some systems may drop echo requests if
                   this header is used.
    -S srcaddr     Source address to use.
    -c compartment Routing compartment identifier.
    -p             Ping a Hyper-V Network Virtualization provider address.
    -4             Force using IPv4.
    -6             Force using IPv6.
```

```
C:\Users\lab1003>ping 142.250.192.132

Pinging 142.250.192.132 with 32 bytes of data:
Reply from 142.250.192.132: bytes=32 time=2ms TTL=57
Reply from 142.250.192.132: bytes=32 time=2ms TTL=57
Reply from 142.250.192.132: bytes=32 time=2ms TTL=57
Reply from 142.250.192.132: bytes=32 time=2ms TTL=57

Ping statistics for 142.250.192.132:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 2ms, Average = 2ms
```

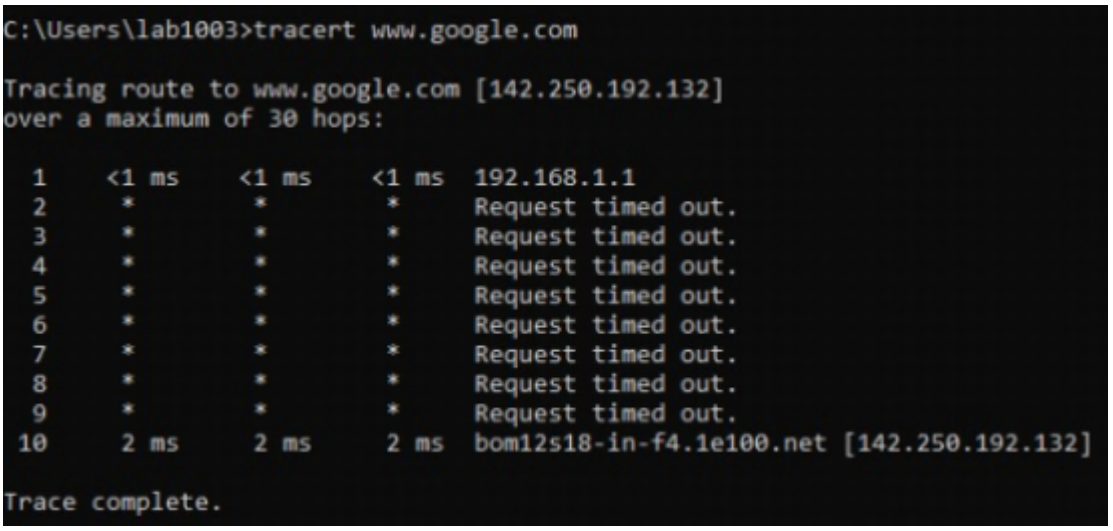
# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

5) **tracert** (windows):
   The Windows **Tracert** tool determines the route to a destination by sending ICMP packets to the destination.
   In these packets, Tracert uses varying IP Time-To-Live (TTL) values.
   The TTL is effectively a hop counter, where a hop is a location that the packet stops at, to reach the destination.[1]
   The tool may take some time to complete (particularly if there is a problem), as the tool waits for responses (which may not come).
   Traceroute is the route tracing tool used on Unix-like Operating Systems

   Parameters:
   Using the -d option with the tracert command instructs TRACERT not to perform a DNS lookup on each IP address, so that TRACERT reports the IP address of the near-side interface of the routers.



```
C:\Users\lab1003>tracert

Usage: tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout]
               [-R] [-S srcaddr] [-4] [-6] target_name

Options:
    -d                 Do not resolve addresses to hostnames.
    -h maximum_hops    Maximum number of hops to search for target.
    -j host-list       Loose source route along host-list (IPv4-only).
    -w timeout         Wait timeout milliseconds for each reply.
    -R                 Trace round-trip path (IPv6-only).
    -S srcaddr         Source address to use (IPv6-only).
    -4                 Force using IPv4.
    -6                 Force using IPv6.
```



```
C:\Users\lab1003>tracert www.google.com

Tracing route to www.google.com [142.250.192.132]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  192.168.1.1
  2     *        *        *     Request timed out.
  3     *        *        *     Request timed out.
  4     *        *        *     Request timed out.
  5     *        *        *     Request timed out.
  6     *        *        *     Request timed out.
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9     *        *        *     Request timed out.
 10     2 ms     2 ms     2 ms  bom12s18-in-f4.1e100.net [142.250.192.132]

Trace complete.
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

6) **nstat** (Unix/Linux):

The **nstat** tool retrieves statistics about the network and SNMP counters from two files, *proc*/net/netstat and *proc*/net/snmp. The format of the contents of the two files is not human-readable, and that's where the nstat command comes in.The netstat command is used to show network status. Traditionally, it is used more for problem determination than for performance measurement. However, the netstat command can be used to determine the amount of traffic on the network to ascertain whether performance problems are due to network congestion
Syntax :netstat [-a] [-b] [-e] [-n] [-o] [-p <Protocol>] [-r] [-s] [<interval>]

```
lab1003@lab1003-HP-280-G2-MT:~$ nstat
#kernel
IpInReceives                    14676              0.0
IpInAddrErrors                  15                 0.0
IpInDelivers                    14645              0.0
IpOutRequests                   13296              0.0
IcmpInMsgs                      156                0.0
IcmpInErrors                    2                  0.0
IcmpInDestUnreachs              35                 0.0
IcmpInTimeExcds                 121                0.0
IcmpMsgInType3                  35                 0.0
IcmpMsgInType11                 121                0.0
TcpActiveOpens                  57                 0.0
TcpEstabResets                  1                  0.0
TcpInSegs                       7429               0.0
TcpOutSegs                      8179               0.0
TcpRetransSegs                  32                 0.0
TcpOutRsts                      24                 0.0
UdpInDatagrams                  6990               0.0
UdpOutDatagrams                 9498               0.0
UdpIgnoredMulti                 41                 0.0
Ip6InReceives                   125                0.0
Ip6InDelivers                   125                0.0
Ip6OutRequests                  1                  0.0
Ip6OutNoRoutes                  974                0.0
Ip6InMcastPkts                  125                0.0
Ip6OutMcastPkts                 1                  0.0
Ip6InOctets                     16876              0.0
Ip6OutOctets                    166                0.0
Ip6InMcastOctets                16876              0.0
Ip6OutMcastOctets               166                0.0
Ip6InNoECTPkts                  125                0.0
Icmp6InMsgs                     1                  0.0
Icmp6InNeighborAdvertisements   1                  0.0
Icmp6InType136                  1                  0.0
Udp6InDatagrams                 124                0.0
Udp6OutDatagrams                1                  0.0
TcpExtTW                        37                 0.0
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

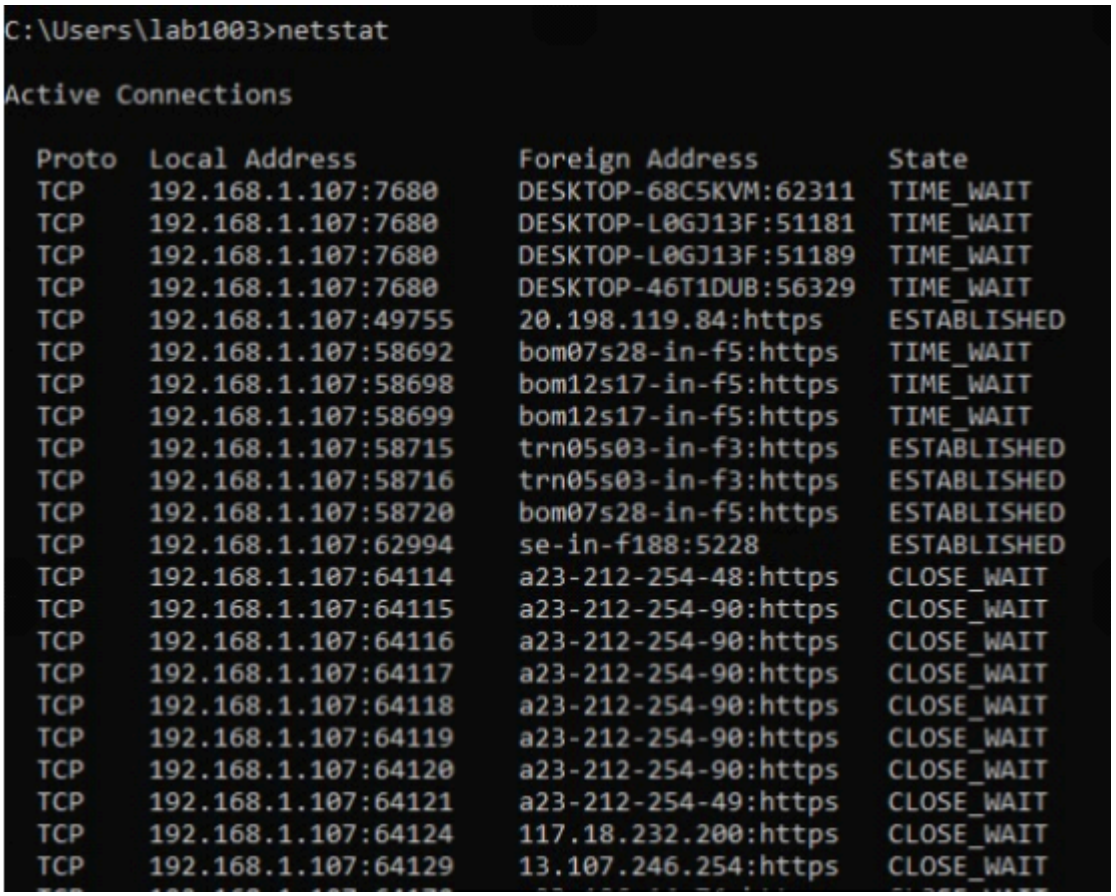
7) **netstat** (Windows) :

The **netstat** command generates displays that show network status and protocol statistics. You can display the status of TCP and UDP endpoints in table format, routing table information, and interface information.

netstat displays various types of network data depending on the command line option selected. These displays are the most useful for system administration. The syntax for this form is:

netstat [-m] [-n] [-s] [-i | -r] [-f address_family]

The most frequently used options for determining network status are: s, r,& i



# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

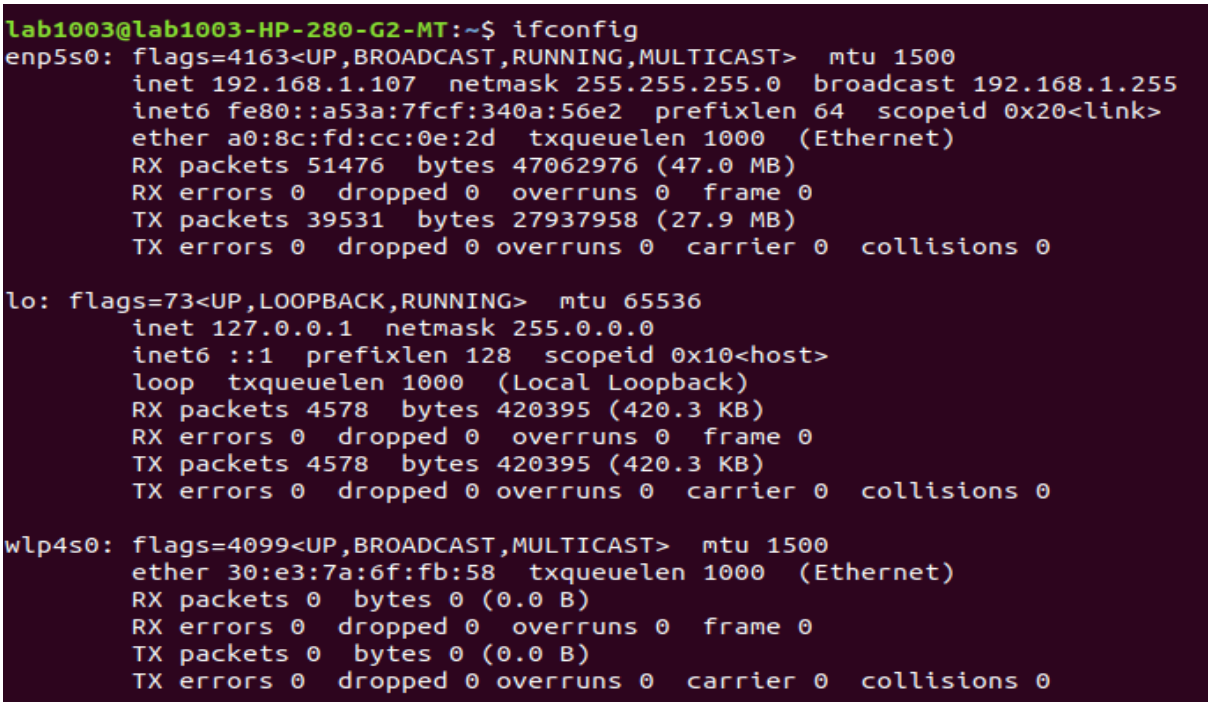
8) **ifconfig** (Unix/Linux):

Configures or displays network interface parameters for a network by using TCP/IP. The **ifconfig** command to assign an address to a network interface and to configure or display the current network interface configuration information. The **ifconfig** command must be used at system startup to define the network address of each interface present on a system. After system startup, it can also be used to redefine an interfaces address and its other operating parameters. The network interface configuration is held on the running system and must be reset at each system restart.

Parameters :

| | |
|---|---|
| **-a** | Optionally, the **-a** flag can be used instead of an interface name. This flag instructs **ifconfig** to display information about all interfaces in the system. |

| | |
|---|---|
| **-help** | Optionally, the **-a** flag can be used instead of an interface name. This flag instructs **ifconfig** to display information about all interfaces in the system. |



***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ ifconfig -v
enp5s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.107  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a53a:7fcf:340a:56e2  prefixlen 64  scopeid 0x20<link>
        ether a0:8c:fd:cc:0e:2d  txqueuelen 1000  (Ethernet)
        RX packets 51722  bytes 47096172 (47.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39723  bytes 28047382 (28.0 MB)
        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 4622  bytes 424111 (424.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4622  bytes 424111 (424.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp4s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 30:e3:7a:6f:fb:58  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ifconfig -a
enp5s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.107  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a53a:7fcf:340a:56e2  prefixlen 64  scopeid 0x20<link>
        ether a0:8c:fd:cc:0e:2d  txqueuelen 1000  (Ethernet)
        RX packets 51534  bytes 47069933 (47.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 39562  bytes 27950878 (27.9 MB)
        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 4582  bytes 420727 (420.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4582  bytes 420727 (420.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp4s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 30:e3:7a:6f:fb:58  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ ifconfig -s
Iface      MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
enp5s0    1500    51676      0      0 0         39713      0      0      0 BMRU
lo       65536     4622      0      0 0          4622      0      0      0 LRU
wlp4s0    1500        0      0      0 0             0      0      0      0 BMU
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ifconfig | grep inet
        inet 192.168.1.107  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a53a:7fcf:340a:56e2  prefixlen 64  scopeid 0x20<link>
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ifconfig -help
Usage:
  ifconfig [-a] [-v] [-s] <interface> [[<AF>] <address>]
  [add <address>[/<prefixlen>]]
  [del <address>[/<prefixlen>]]
  [[-]broadcast [<address>]]  [[-]pointopoint [<address>]]
  [netmask <address>]  [dstaddr <address>]  [tunnel <address>]
  [outfill <NN>] [keepalive <NN>]
  [hw <HW> <address>]  [mtu <NN>]
  [[-]trailers]  [[-]arp]  [[-]allmulti]
  [multicast]  [[-]promisc]
  [mem_start <NN>]  [io_addr <NN>]  [irq <NN>]  [media <type>]
  [txqueuelen <NN>]
  [[-]dynamic]
  [up|down] ...

  <HW>=Hardware Type.
  List of possible hardware types:
    loop (Local Loopback) slip (Serial Line IP) cslip (VJ Serial Line IP)
    slip6 (6-bit Serial Line IP) cslip6 (VJ 6-bit Serial Line IP) adaptive (Adaptive Serial Line IP)
    ash (Ash) ether (Ethernet) ax25 (AMPR AX.25)
    netrom (AMPR NET/ROM) rose (AMPR ROSE) tunnel (IPIP Tunnel)
    ppp (Point-to-Point Protocol) hdlc ((Cisco)-HDLC) lapb (LAPB)
    arcnet (ARCnet) dlci (Frame Relay DLCI) frad (Frame Relay Access Device)
    sit (IPv6-in-IPv4) fddi (Fiber Distributed Data Interface) hippi (HIPPI)
    irda (IrLAP) ec (Econet) x25 (generic X.25)
    eui64 (Generic EUI-64)
  <AF>=Address family. Default: inet
  List of possible address families:
    unix (UNIX Domain) inet (DARPA Internet) inet6 (IPv6)
    ax25 (AMPR AX.25) netrom (AMPR NET/ROM) rose (AMPR ROSE)
    ipx (Novell IPX) ddp (Appletalk DDP) ec (Econet)
    ash (Ash) x25 (CCITT X.25)
```

9) **ip** (Unix/Linux) :

**Ip** command in Linux is present in the net-tools which are used for performing several network administration tasks. IP stands for Internet Protocol. This command is used to show or manipulate routing, devices, and tunnels.
The ip command is used to perform several tasks like assigning an address to a network interface or configuring network interface parameters. It can perform several other tasks like configuring and modifying the default and static routing, setting up a tunnel over IP, listing IP addresses and property information, modifying the status of the interface, and assigning, deleting, and setting up IP addresses and routes.

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

Syntax : ip [ OPTIONS ] OBJECT { COMMAND | help }
Parameters:
**-address** : This option is used to show all IP addresses associated with all network devices.
**-link** : It is used to display link layer information; it will fetch characteristics of the link layer devices currently available. Any networking device which has a driver loaded can be classified as an available device.
**-s -link** : This link option when used with -s option is used to show the statistics of the various network interfaces.
**-route** : This command helps you to see the route packets your network will take as set in your routing table. The first entry is the default route.

```
lab1003@lab1003-HP-280-G2-MT:~$ ip
Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }
       ip [ -force ] -batch filename
where  OBJECT := { link | address | addrlabel | route | rule | neigh | ntable |
                   tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm |
                   netns | l2tp | fou | macsec | tcp_metrics | token | netconf | ila |
                   vrf | sr }
       OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |
                    -h[uman-readable] | -iec |
                    -f[amily] { inet | inet6 | ipx | dnet | mpls | bridge | link } |
                    -4 | -6 | -I | -D | -B | -0 |
                    -l[oops] { maximum-addr-flush-attempts } | -br[ief] |
                    -o[neline] | -t[imestamp] | -ts[hort] | -b[atch] [filename] |
                    -rc[vbuf] [size] | -n[etns] name | -a[ll] | -c[olor]}
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ip address
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
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp5s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether a0:8c:fd:cc:0e:2d brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.107/24 brd 192.168.1.255 scope global dynamic noprefixroute enp5s0
       valid_lft 4550sec preferred_lft 4550sec
    inet6 fe80::a53a:7fcf:340a:56e2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: wlp4s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default qlen 1000
    link/ether 30:e3:7a:6f:fb:58 brd ff:ff:ff:ff:ff:ff
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ip route
default via 192.168.1.1 dev enp5s0 proto dhcp metric 100
169.254.0.0/16 dev enp5s0 scope link metric 1000
192.168.1.0/24 dev enp5s0 proto kernel scope link src 192.168.1.107 metric 100
lab1003@lab1003-HP-280-G2-MT:~$ ip neighbour
192.168.1.1 dev enp5s0 lladdr 10:27:f5:a9:23:47 REACHABLE
lab1003@lab1003-HP-280-G2-MT:~$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp5s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether a0:8c:fd:cc:0e:2d brd ff:ff:ff:ff:ff:ff
3: wlp4s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DORMANT group default qlen 1000
    link/ether 30:e3:7a:6f:fb:58 brd ff:ff:ff:ff:ff:ff
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

```
lab1003@lab1003-HP-280-G2-MT:~$ ip -s link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    RX: bytes  packets  errors  dropped overrun mcast
    455785     4969     0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    455785     4969     0       0       0       0
2: enp5s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether a0:8c:fd:cc:0e:2d brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    47780476   53934    0       0       0       2004
    TX: bytes  packets  errors  dropped carrier collsns
    28604276   40978    0       0       0       0
3: wlp4s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN mode DORMANT group default qlen 1000
    link/ether 30:e3:7a:6f:fb:58 brd ff:ff:ff:ff:ff:ff
    RX: bytes  packets  errors  dropped overrun mcast
    0          0        0       0       0       0
    TX: bytes  packets  errors  dropped carrier collsns
    0          0        0       0       0       0
```

10)**traceroute** (Unix/Linux) :

**traceroute** command in Linux prints the route that a packet takes to reach the host. This command is useful when you want to know about the route and about all the hops that a packet takes. Below image depicts how traceroute command is used to reach the Google host from the local machine and it also prints detail about all the hops that it visits in between.
Syntax : traceroute [options]  host_Address [pathlength]
Parameters :
**-4 Option:** Use ip version 4 i.e. use IPv4
**-6 Option:** Use ip version 6 i.e. use IPv6
**-F Option:** Do not fragment packet
**–help**: Display help messages and exit.

```
lab1003@lab1003-HP-280-G2-MT:~$ traceroute www.google.com
traceroute to www.google.com (142.250.192.132), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.1)  0.319 ms  0.250 ms  0.225 ms
 2  Archer (192.168.0.1)  2.084 ms  1.993 ms  1.988 ms
 3  203.212.25.1 (203.212.25.1)  2.312 ms  2.263 ms  2.330 ms
 4  * 203.212.24.53 (203.212.24.53)  2.840 ms  2.876 ms
 5  175.100.177.53 (175.100.177.53)  3.237 ms  3.273 ms  4.201 ms
 6  * * *
 7  175.100.188.22 (175.100.188.22)  2.832 ms  2.718 ms  2.806 ms
 8  * * *
 9  74.125.253.164 (74.125.253.164)  5.213 ms 142.250.228.50 (142.250.228.50)  4.911 ms 74.125.253.166 (74.125.253.166)  3.535 ms
10  172.253.50.147 (172.253.50.147)  3.488 ms 142.250.209.70 (142.250.209.70)  9.107 ms 142.250.238.81 (142.250.238.81)  10.839 ms
11  bom12s18-in-f4.1e100.net (142.250.192.132)  117.923 ms  117.871 ms 108.170.248.177 (108.170.248.177)  4.370 ms
```

```
lab1003@lab1003-HP-280-G2-MT:~$ traceroute -4 google.com
traceroute to google.com (142.251.42.46), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.1)  0.213 ms  0.171 ms  0.155 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * 142.250.60.134 (142.250.60.134)  3.490 ms 142.250.227.70 (142.250.227.70)  3.572 ms
10  142.250.226.134 (142.250.226.134)  3.980 ms 142.251.69.43 (142.251.69.43)  4.199 ms 142.251.69.45 (142.251.69.45)  3.532 ms
11  bom12s20-in-f14.1e100.net (142.251.42.46)  4.122 ms  4.072 ms 108.170.248.161 (108.170.248.161)  4.230 ms
```

***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ traceroute -6 google.com
traceroute to google.com (2404:6800:4009:830::200e), 30 hops max, 80 byte packets
connect: Network is unreachable
```

```
lab1003@lab1003-HP-280-G2-MT:~$ traceroute -V google.com
Modern traceroute for Linux, version 2.1.0
```

```
lab1003@lab1003-HP-280-G2-MT:~$ traceroute -e google.com
traceroute to google.com (142.251.42.46), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.1)  0.814 ms  0.769 ms  0.731 ms
 2  Archer (192.168.0.1)  1.924 ms  2.100 ms  2.051 ms
 3  203.212.25.1 (203.212.25.1)  2.350 ms  2.346 ms  2.306 ms
 4  203.212.24.53 (203.212.24.53)  2.267 ms  2.232 ms  2.236 ms
 5  175.100.177.53 (175.100.177.53)  3.655 ms  3.637 ms  3.697 ms
 6  * * *
 7  175.100.188.22 (175.100.188.22)  3.060 ms  2.144 ms  2.161 ms
 8  * * *
 9  142.251.69.102 (142.251.69.102)  6.353 ms 172.253.77.22 (172.253.77.22)  5.053 ms 142.250.227.70 (142.250.227.70)  3.745 ms
10  142.250.209.70 (142.250.209.70)  3.835 ms 108.170.248.194 (108.170.248.194)  4.791 ms 142.250.208.226 (142.250.208.226)  3.737 ms
11  108.170.248.177 (108.170.248.177)  4.922 ms  4.846 ms bom12s20-in-f14.1e100.net (142.251.42.46)  2.082 ms
```

```
lab1003@lab1003-HP-280-G2-MT:~$ traceroute -F google.com
traceroute to google.com (142.251.42.46), 30 hops max, 60 byte packets
 1  _gateway (192.168.1.1)  0.258 ms  0.194 ms  0.152 ms
 2  Archer (192.168.0.1)  1.454 ms  1.799 ms  1.899 ms
 3  203.212.25.1 (203.212.25.1)  2.111 ms  2.059 ms  2.007 ms
 4  203.212.24.53 (203.212.24.53)  1.956 ms  1.982 ms  1.929 ms
 5  175.100.177.53 (175.100.177.53)  8.322 ms  3.373 ms  3.335 ms
 6  * 172.16.2.202 (172.16.2.202)  17.326 ms *
 7  175.100.188.22 (175.100.188.22)  2.275 ms  2.981 ms  3.018 ms
 8  * * *
 9  74.125.251.132 (74.125.251.132)  5.899 ms 216.239.54.146 (216.239.54.146)  5.425 ms 142.250.214.100 (142.250.214.100)  5.377 ms
10  142.251.69.45 (142.251.69.45)  2.845 ms  5.310 ms 142.251.69.43 (142.251.69.43)  3.662 ms
11  108.170.248.161 (108.170.248.161)  3.327 ms  3.265 ms 108.170.248.177 (108.170.248.177)  3.387 ms
12  bom12s20-in-f14.1e100.net (142.251.42.46)  2.153 ms 142.251.69.43 (142.251.69.43)  3.129 ms bom12s20-in-f14.1e100.net (142.251.42.46)  2.0
50 ms
```

11) **tracepath** (Unix/Linux):

**tracepath** command in Linux is used to traces path to destination discovering MTU along this path. It uses UDP port or some random port. It is similar to traceroute, but it does not require superuser privileges and has no fancy options. tracepath6 is a good replacement for traceroute6 and classic example of the application of Linux error queues. The situation with IPv4 is worse because commercial IP routers do not return enough information in ICMP error messages. Probably, it will change, when they will be updated. For now, it uses Van Jacobson's trick, sweeping a range of UDP ports to maintain trace history.

Syntax : tracepath [-n] [-b] [-l pktlen] [-m max_hops] [-p port] destination
Parameters : tracepath command without any option: It will print the general syntax of the command along with the various options that can be used with the tracepath command as well as gives a brief description about each option
**tracepath -n:** This option prints primarily IP addresses numerically

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

**tracepath -l** : This option sets the initial packet length to pktlen instead of 65535 for tracepath or 128000 for tracepath6.
**tracepath -p:** This option will set the initial destination port to use.

```
lab1003@lab1003-HP-280-G2-MT:~$ tracepath
Usage: tracepath [-n] [-b] [-l <len>] [-p port] <destination>
```

```
lab1003@lab1003-HP-280-G2-MT:~$ tracepath -p 8080 www.google.com
 1?: [LOCALHOST]                      pmtu 1500
 1:  _gateway                                              1.034ms
 1:  _gateway                                              0.632ms
 2:  no reply
 3:  no reply
 4:  no reply
 5:  no reply
 6:  no reply
```

```
lab1003@lab1003-HP-280-G2-MT:~$ tracepath -n www.google.com
 1?: [LOCALHOST]                      pmtu 1500
 1:  192.168.1.1                                           0.666ms
 1:  192.168.1.1                                           0.619ms
 2:  no reply
 3:  no reply
 4:  no reply
 5:  no reply
 6:  no reply
 7:  no reply
 8:  no reply
 9:  no reply
10:  no reply
11:  no reply
12:  no reply
```

```
lab1003@lab1003-HP-280-G2-MT:~$ tracepath -l  29 www.google.com
 1:  _gateway                                              0.364ms
 2:  no reply
 3:  no reply
 4:  no reply
 5:  no reply
```

```
lab1003@lab1003-HP-280-G2-MT:~$ tracepath -b www.google.com
 1?: [LOCALHOST]                      pmtu 1500
 1:  _gateway (192.168.1.1)                                0.623ms
 1:  _gateway (192.168.1.1)                                0.570ms
 2:  no reply
 3:  no reply
 4:  no reply
 5:  no reply
^C
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

12) **ss** (Unix/Linux):

The **ss** command is a tool used to dump socket statistics and displays information in similar fashion (although simpler and faster) to netstat. The ss command can also display even more TCP and state information than most other tools.'ss' command, which stands for "Socket Statistics." It is a potent tool for inspecting and displaying detailed information about network sockets on a Linux system. The 'ss' command is an indispensable resource for network administrators, system administrators, and developers, offering insights into network connections, routing tables, and more.

Syntax:ss [options]
Parameters :
**-t** Display TCP sockets
**-u** Display UDP sockets
**-l** Display listening sockets

```
lab1003@lab1003-HP-280-G2-MT:~$ ss
Netid      State       Recv-Q      Send-Q                                Local Address:Port                 Peer Address:Port
u_str      ESTAB       0           0                                  /run/user/1000/bus 38160                 * 39222
u_str      ESTAB       0           0                                                     * 37016                 * 39072
u_str      ESTAB       0           0                                                     * 31836                 * 30920
u_str      ESTAB       0           0                                                     * 25232                 * 23456
u_str      ESTAB       0           0                  /var/run/dbus/system_bus_socket 37475                 * 38492
u_str      ESTAB       0           0                                                     * 34551                 * 35169
u_str      ESTAB       0           0                                                     * 43619                 * 44108
u_str      ESTAB       0           0                                                     * 29621                 * 29622
u_str      ESTAB       0           0                                                     * 39014                 * 36986
u_str      ESTAB       0           0                                                     * 33631                 * 35167
u_str      ESTAB       0           0                     /run/systemd/journal/stdout 36366                 * 33571
u_str      ESTAB       0           0                                                     * 30916                 * 30917
u_str      ESTAB       0           0                               /run/user/121/bus 29864                 * 28327
u_str      ESTAB       0           0                                                     * 99127                 * 99126
u_str      ESTAB       0           0                          @/tmp/dbus-DwJidvMdaW 68944                 * 66383
u_str      ESTAB       0           0                                                     * 39222                 * 38160
u_str      ESTAB       0           0                                  /run/user/1000/bus 39128                 * 37065
u_str      ESTAB       0           0                  /var/run/dbus/system_bus_socket 28624                 * 32445
u_str      ESTAB       0           0                                                     * 28954                 * 26799
u_seq      ESTAB       0           0                                                     * 45253                 * 45252
u_str      ESTAB       0           0                                  /run/user/1000/bus 34396                 * 33572
u_str      ESTAB       0           0                                                     * 23678                 * 24673
u_str      ESTAB       0           0                     /run/systemd/journal/stdout 44108                 * 43619
u_str      ESTAB       0           0                                                     * 29643                 * 29842
u_seq      ESTAB       0           0                                                     * 45644                 * 45645
u_str      ESTAB       0           0                                                     * 39035                 * 38012
u_str      ESTAB       0           0                                  /run/user/1000/bus 35167                 * 33631
u_str      ESTAB       0           0                          @/tmp/dbus-HQ9wF8ppCm 30905                 * 30904
u_str      ESTAB       0           0                               /run/user/121/bus 30871                 * 28305
u_str      ESTAB       0           0                                                     * 26936                 * 29371
u_seq      ESTAB       0           0                                                     * 92176                 * 0
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ss -t
State          Recv-Q          Send-Q                   Local Address:Port                    Peer Address:Port
ESTAB          0               0                      192.168.1.107:36764                  142.250.182.197:https
ESTAB          0               0                      192.168.1.107:42854                   172.217.166.78:https
ESTAB          0               0                      192.168.1.107:49480                    34.120.5.221:https
ESTAB          0               0                      192.168.1.107:34562                  142.250.67.138:https
ESTAB          0               0                      192.168.1.107:59820                   34.107.243.93:https
```

```
lab1003@lab1003-HP-280-G2-MT:~$ ss -u
Recv-Q          Send-Q                   Local Address:Port                    Peer Address:Port
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

13) **dig** (Unix/Linux):

**dig** command stands for **Domain Information Groper.** It is used for retrieving information about DNS name servers. It is basically used by network administrators. It is used for verifying and troubleshooting DNS problems and to perform DNS lookups. Dig command replaces older tools such as nslookup and the host

Syntax : dig [server] [name] [type]
Parameters :
**dig [server name] [ANY]** We use “ANY” option to query all the available DNS record types associated with a domain. It will include all the available record types in the output. 7. To query MX record for the domain

```
lab1003@lab1003-HP-280-G2-MT:~$ dig

; <<>> DiG 9.11.3-1ubuntu1.18-Ubuntu <<>>
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 27016
;; flags: qr rd ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;.                              IN      NS

;; ANSWER SECTION:
.                       737     IN      NS      g.root-servers.net.
.                       737     IN      NS      h.root-servers.net.
.                       737     IN      NS      i.root-servers.net.
.                       737     IN      NS      j.root-servers.net.
.                       737     IN      NS      k.root-servers.net.
.                       737     IN      NS      l.root-servers.net.
.                       737     IN      NS      m.root-servers.net.
.                       737     IN      NS      a.root-servers.net.
.                       737     IN      NS      b.root-servers.net.
.                       737     IN      NS      c.root-servers.net.
.                       737     IN      NS      d.root-servers.net.
.                       737     IN      NS      e.root-servers.net.
.                       737     IN      NS      f.root-servers.net.

;; Query time: 3 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Thu Jan 25 11:54:39 IST 2024
;; MSG SIZE  rcvd: 239
```

***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ dig yahoo.com ANY

; <<>> DiG 9.11.3-1ubuntu1.18-Ubuntu <<>> yahoo.com ANY
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 56581
;; flags: qr rd ra; QUERY: 1, ANSWER: 6, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 65494
;; QUESTION SECTION:
;yahoo.com.                     IN      ANY

;; ANSWER SECTION:
yahoo.com.              1159    IN      A       74.6.231.21
yahoo.com.              1159    IN      A       74.6.143.26
yahoo.com.              1159    IN      A       74.6.143.25
yahoo.com.              1159    IN      A       74.6.231.20
yahoo.com.              1159    IN      A       98.137.11.164
yahoo.com.              1159    IN      A       98.137.11.163

;; Query time: 3 msec
;; SERVER: 127.0.0.53#53(127.0.0.53)
;; WHEN: Thu Jan 25 11:54:26 IST 2024
;; MSG SIZE  rcvd: 134
```

14) **ssh** (Unix/Linux):

**Secure Shell (SSH)** is a protocol for secure remote access and other secure network services over an insecure network. It is widely used by system administrators and developers to securely manage remote servers and perform other network operations.
The **ssh** command is used to securely log into a remote machine and execute commands on that machine. The basic syntax of the command is “ssh user@host”, where user is the username on the remote machine and host is the address or hostname of the remote machine.

```
C:\Users\lab1003>SSH
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command]
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

15) ***route*** (windows) :

The **route** command allows you to make manual entries into the network routing tables. The **route** command distinguishes between routes to hosts and routes to networks by interpreting the network address of the *Destination* variable, which can be specified either by symbolic name or numeric address. The **route** command resolves all symbolic names into addresses, using either the file or the network name server.Routes to a particular host are distinguished from those to a network by interpreting the Internet address associated with the destination. The optional phs **-net** and **-host** force the destination to be interpreted as a network or a host, respectively. If the destination has a local address part of INADDR_ANY or if the destination is the symbolic name of a network, then the route is assumed to be to a network; otherwise, it is presumed to be a route to a host.

Parameters :

| | |
|---|---|
| **-net** | Indicates that the *Destination* parameter should be interpreted as a network. |
| **-netmask** | Specifies the network mask to the destination address. Make sure this option follows the *Destination* parameter. |
| **-host** | Indicates that the *Destination* parameter should be interpreted as a host. |

***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
C:\Users\lab1003>route -p

Manipulates network routing tables.

ROUTE [-f] [-p] [-4|-6] command [destination]
                  [MASK netmask]  [gateway] [METRIC metric]  [IF interface]

  -f           Clears the routing tables of all gateway entries.  If this is
               used in conjunction with one of the commands, the tables are
               cleared prior to running the command.

  -p           When used with the ADD command, makes a route persistent across
               boots of the system. By default, routes are not preserved
               when the system is restarted. Ignored for all other commands,
               which always affect the appropriate persistent routes.

  -4           Force using IPv4.

  -6           Force using IPv6.

  command      One of these:
                 PRINT     Prints  a route
                 ADD       Adds    a route
                 DELETE    Deletes a route
                 CHANGE    Modifies an existing route
  destination  Specifies the host.
  MASK         Specifies that the next parameter is the 'netmask' value.
  netmask      Specifies a subnet mask value for this route entry.
               If not specified, it defaults to 255.255.255.255.
  gateway      Specifies gateway.
  interface    the interface number for the specified route.
  METRIC       specifies the metric, ie. cost for the destination.

All symbolic names used for destination are looked up in the network database
file NETWORKS. The symbolic names for gateway are looked up in the host name
database file HOSTS.
```

16) **host** (Unix/Linux):

The **host** command is a command-line tool that performs DNS (Domain Name System) lookups, The tool can find a hostname when provided with an IP address. However, the command returns an IP address when given a hostname.

Syntax : host [options] [domain hostname IP address]
Parameters :
host [hostname]
host [IP address]
host [domain]

```
lab1003@lab1003-HP-280-G2-MT:~$ host www.google.com
www.google.com has address 142.250.192.132
www.google.com has IPv6 address 2404:6800:4009:82b::2004
```

***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ host
Usage: host [-aCdilrTvVw] [-c class] [-N ndots] [-t type] [-W time]
            [-R number] [-m flag] hostname [server]
       -a is equivalent to -v -t ANY
       -c specifies query class for non-IN data
       -C compares SOA records on authoritative nameservers
       -d is equivalent to -v
       -i IP6.INT reverse lookups
       -l lists all hosts in a domain, using AXFR
       -m set memory debugging flag (trace|record|usage)
       -N changes the number of dots allowed before root lookup is done
       -r disables recursive processing
       -R specifies number of retries for UDP packets
       -s a SERVFAIL response should stop query
       -t specifies the query type
       -T enables TCP/IP mode
       -v enables verbose output
       -V print version number and exit
       -w specifies to wait forever for a reply
       -W specifies how long to wait for a reply
       -4 use IPv4 query transport only
       -6 use IPv6 query transport only
```

17) ***arp*** (windows) :

**arp** command manipulates the System's ARP cache. It also allows a complete dump of the ARP cache. ARP stands for **Address Resolution Protocol.** The primary function of this protocol is to resolve the IP address of a system to its mac address, and hence it works between level 2(Data link layer) and level 3(Network layer).

Syntax : arp [-v] [-i if] [-H type] -a [hostname]
Parameters :
**-a [hostname] –all:** This option is used for showing entries of the specified host. If nothing is passed all entries will be displayed.
**-v, –verbose**: This option shows the verbose information.
**-n, –numeric:** This option shows numerical addresses instead of symbolic host, port or usernames.

***KUNAL NEMADE***

***ROLL NO :. 89***

***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
C:\Users\lab1003>arp

Displays and modifies the IP-to-Physical address translation tables used by
address resolution protocol (ARP).

ARP -s inet_addr eth_addr [if_addr]
ARP -d inet_addr [if_addr]
ARP -a [inet_addr] [-N if_addr] [-v]

  -a            Displays current ARP entries by interrogating the current
                protocol data.  If inet_addr is specified, the IP and Physical
                addresses for only the specified computer are displayed.  If
                more than one network interface uses ARP, entries for each ARP
                table are displayed.
  -g            Same as -a.
  -v            Displays current ARP entries in verbose mode.  All invalid
                entries and entries on the loop-back interface will be shown.
  inet_addr     Specifies an internet address.
  -N if_addr    Displays the ARP entries for the network interface specified
                by if_addr.
  -d            Deletes the host specified by inet_addr. inet_addr may be
                wildcarded with * to delete all hosts.
  -s            Adds the host and associates the Internet address inet_addr
                with the Physical address eth_addr.  The Physical address is
                given as 6 hexadecimal bytes separated by hyphens. The entry
                is permanent.
  eth_addr      Specifies a physical address.
  if_addr       If present, this specifies the Internet address of the
                interface whose address translation table should be modified.
                If not present, the first applicable interface will be used.
Example:
  > arp -s 157.55.85.212   00-aa-00-62-c6-09  .... Adds a static entry.
  > arp -a                                    .... Displays the arp table.
```

```
C:\Users\lab1003>arp -a

Interface: 192.168.1.107 --- 0xc
  Internet Address      Physical Address      Type
  192.168.1.1           10-27-f5-a9-23-47     dynamic
  192.168.1.101         a0-8c-fd-c4-b6-d4     dynamic
  192.168.1.104         f4-39-09-49-0a-33     dynamic
  192.168.1.106         a0-8c-fd-d9-9f-1a     dynamic
  192.168.1.108         04-0e-3c-1a-62-37     dynamic
  192.168.1.109         f4-39-09-48-b0-56     dynamic
  192.168.1.112         04-0e-3c-1a-66-09     dynamic
  192.168.1.113         04-0e-3c-19-28-92     dynamic
  192.168.1.118         f4-39-09-49-6c-ab     dynamic
  192.168.1.119         a0-8c-fd-cc-0d-43     dynamic
  192.168.1.120         a0-8c-fd-da-1f-44     dynamic
  192.168.1.121         f4-39-09-49-6c-e3     dynamic
  192.168.1.132         04-0e-3c-1a-5c-c8     dynamic
  192.168.1.134         04-0e-3c-19-28-a2     dynamic
  192.168.1.135         04-0e-3c-1a-64-e4     dynamic
  192.168.1.136         a0-8c-fd-cc-0e-1a     dynamic
  192.168.1.138         f4-39-09-49-6c-ec     dynamic
  192.168.1.141         f4-39-09-49-6c-fc     dynamic
  192.168.1.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.2             01-00-5e-00-00-02     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

18) ***hostname*** (windows) :

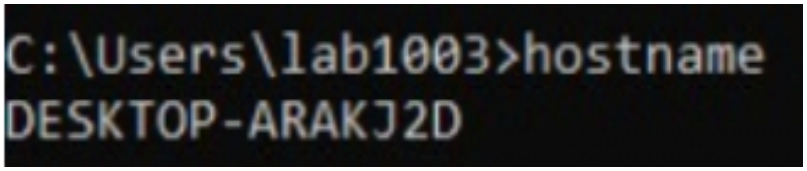
Sets or displays the name of the current host system
Hostname is used to display the system's DNS name, and to display or set its hostname or NIS domain name

**GET NAME:** When called without any arguments, the program displays the current names: hostname will print the name of the system as returned by the gethostname(2) function.
**SET NAME:** When called with one argument or with the --file option, the commands set the host name or the NIS/YP domain name. hostname uses the sethostname(2) function, while all of the three domainname, ypdomainname and nisdomainname use setdomainname(2). Note, that this is effective only until the next reboot. Edit /etc/hostname for permanent change

Syntax : hostname [-a|--alias] [-d|--domain] [-f|--fqdn|--long] [-A|--all-fqdns] [-i|--ip-address] [-I|--all-ip-addresses] [-s|--short] [-y|--yp|--nis]
hostname [-b|--boot] [-F|--file filename] [hostname]
hostname [-h|--help] [-V|--version

Parameters :

| | |
|---|---|
| **-s** | Trims any domain information from the printed name. |

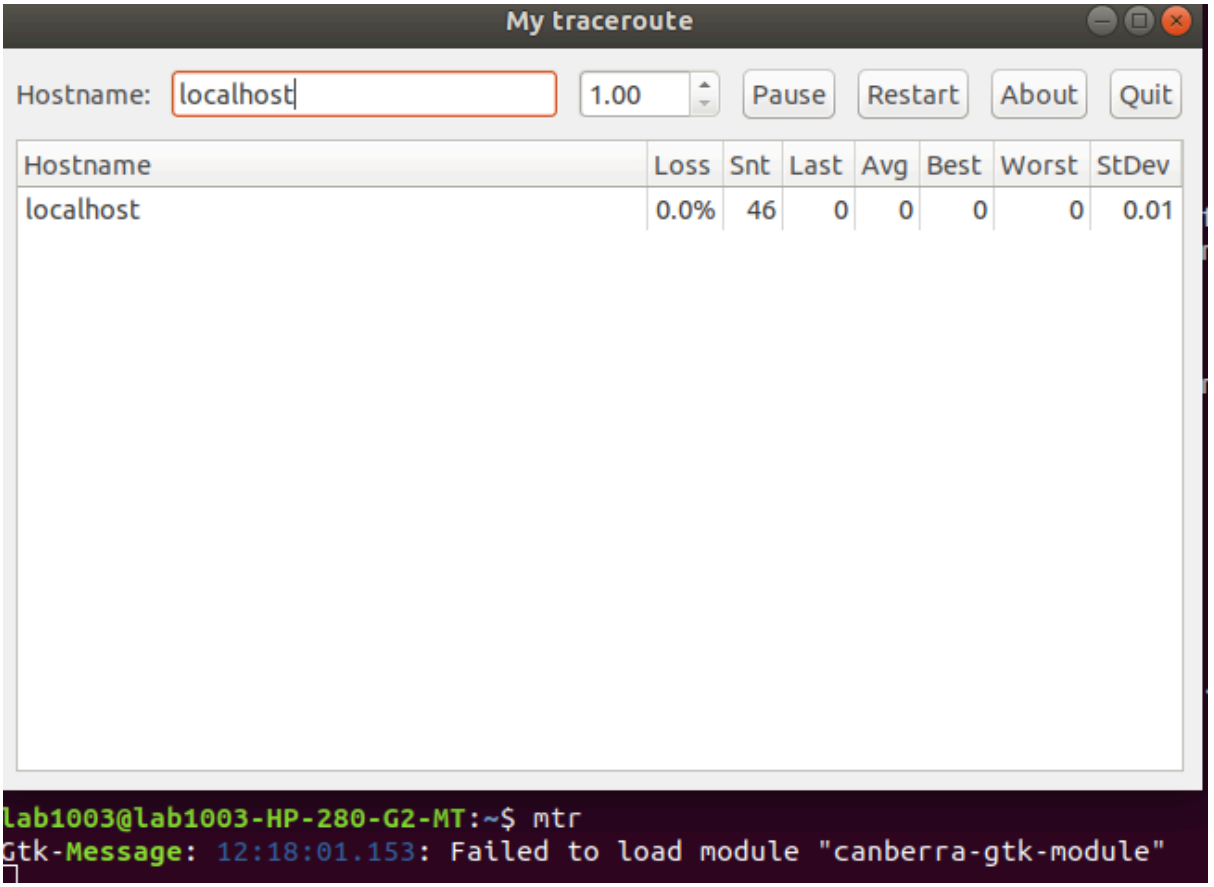



19) **mtr** (Unix/Linux):

The name is a shorthand for My Traceroute, also known as Matt's Traceroute. mtr is a networking tool that combines ping and traceroute to diagnose a network. Instead of using both tools separately, we could use only mtr. The purpose of mtr is to  analyze the network traffic hop-to-hop using ICMP packetsThe mtr command is a combination of ping and traceroute commands. It is a network diagnostic tool that continuously sends packets

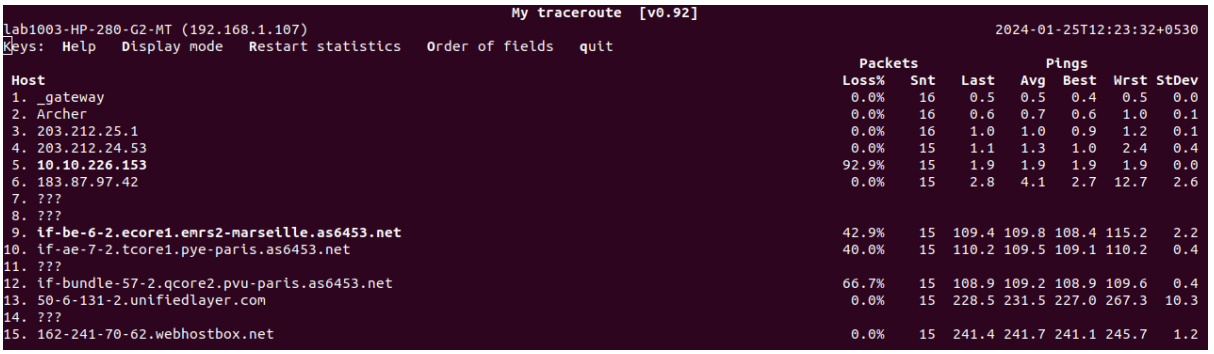
showing ping time for each hop. It also displays network problems of the entire route taken by the network packets.

Syntax : mtr <option> <hostname>/path
Parameters :
**-t, --curses:** It is used to specify the use of curses-based terminal interface forcefully.
**-v, --version:** It is used to display the installed version information.
**-4:** It is used for IPv4 addresses.
**-6:** It is used for IPv6 addresses.




```
lab1003@lab1003-HP-280-G2-MT:~$ mtr -t tsec.edu
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*




20) **whois** (Unix/Linux) :

The whois command displays information about a website's record. You may get all the information about a website regarding its registration and owner's information , Identifies a user by user ID or alias.The whois command tries to reach ARPANET host internic.net where it examines a user-name database to obtain information. The whois command should be used only by users on ARPANET. Refer to RFC 812 for more complete information and recent changes to the whois command.

Syntax : whois **<websiteName>**
Parameters :

| | |
|---|---|
| **.** | Forces a name-only search for the name specified in the *Name* parameter. |
| **!** | Displays help information for the nickname or handle ID specified in the *Name* parameter. |
| ***** | Displays the entire membership list of a group or organization. If there are many members, this can take some time. |
| **?** | Requests help from the ARPANET host. |
| **-h** *HostName* | Specifies an alternative host name. The default host name on the ARPANET is internic.net. You can contact the other major ARPANET user-name database, nic.ddn.mil, by specifying the **-h** *HostName* flag. |

***KUNAL NEMADE***

***ROLL NO :. 89***

***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ whois tsec.edu
This Registry database contains ONLY .EDU domains.
The data in the EDUCAUSE Whois database is provided
by EDUCAUSE for information purposes in order to
assist in the process of obtaining information about
or related to .edu domain registration records.

The EDUCAUSE Whois database is authoritative for the
.EDU domain.

A Web interface for the .EDU EDUCAUSE Whois Server is
available at: http://whois.educause.edu

By submitting a Whois query, you agree that this information
will not be used to allow, enable, or otherwise support
the transmission of unsolicited commercial advertising or
solicitations via e-mail.  The use of electronic processes to
harvest information from this server is generally prohibited
except as reasonably necessary to register or modify .edu
domain names.
```

```
Domain Name: TSEC.EDU

Registrant:
        Thadomal Sahani Engineering College
        P.G Kher Marg, Bandra(W)
        Mumbai, Maharashtra 400 050
        India

Administrative Contact:
        Dr. Gopakumaran Thampi
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
        India
        +91.2226495808
        gtthampi@yahoo.com

Technical Contact:
        Chetan Agarwal
        Thadomal Shahani Engineering College
        Nari Gurshahani Marg, Bandra(W)
        Mumbai, 400050
        India
        +91.2226495808
        chetan.agarwal@thadomal.org

Name Servers:
        NS2.SALESUPP.IN
        NS1.SALESUPP.IN

Domain record activated:    22-Jan-2001
Domain record last updated: 31-Aug-2023
Domain expires:             31-Jul-2024
```

***KUNAL NEMADE***
***ROLL NO :. 89***
***NL LAB SE/S2/89/ASSIGNMENT NO :. 01***

```
lab1003@lab1003-HP-280-G2-MT:~$ whois -i www.facebook.com
Usage: whois [OPTION]... OBJECT...

-h HOST, --host HOST   connect to server HOST
-p PORT, --port PORT   connect to PORT
-H                     hide legal disclaimers
      --verbose        explain what is being done
      --help           display this help and exit
      --version        output version information and exit

These flags are supported by whois.ripe.net and some RIPE-like servers:
-l                     find the one level less specific match
-L                     find all levels less specific matches
-m                     find all one level more specific matches
-M                     find all levels of more specific matches
-c                     find the smallest match containing a mnt-irt attribute
-x                     exact match
-b                     return brief IP address ranges with abuse contact
-B                     turn off object filtering (show email addresses)
-G                     turn off grouping of associated objects
-d                     return DNS reverse delegation objects too
-i ATTR[,ATTR]...      do an inverse look-up for specified ATTRibutes
-T TYPE[,TYPE]...      only look for objects of TYPE
-K                     only primary keys are returned
-r                     turn off recursive look-ups for contact information
-R                     force to show local copy of the domain object even
                       if it contains referral
-a                     also search all the mirrored databases
-s SOURCE[,SOURCE]...  search the database mirrored from SOURCE
-g SOURCE:FIRST-LAST   find updates from SOURCE from serial FIRST to LAST
-t TYPE                request template for object of TYPE
-v TYPE                request verbose template for object of TYPE
-q [version|sources|types]  query specified server info
```

21) **tcpdump** (Unix/Linux) :

**tcpdump** is a packet sniffing and packet analyzing tool for a System Administrator to troubleshoot connectivity issues in Linux. It is used to capture, filter, and analyze network traffic such as TCP/IP packets going through your system. It is many times used as a security tool as well. It saves the captured information in a pcap file, these pcap files can then be opened through Wireshark or through the command tool itself.

Syntax : sudo tcpdump
Parameter : To capture the packets of current network interface

# KUNAL NEMADE
# ROLL NO :. 89
# NL LAB SE/S2/89/ASSIGNMENT NO :. 01

```
lab1003@lab1003-HP-280-G2-MT:~$ sudo tcpdump
[sudo] password for lab1003:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp5s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:51:15.850787 ARP, Request who-has 192.168.1.100 tell _gateway, length 46
11:51:15.850905 ARP, Request who-has 192.168.1.133 tell _gateway, length 46
11:51:15.852395 IP lab1003-HP-280-G2-MT.40288 > _gateway.domain: 20966+ PTR? 100.1.168.192.in-addr.arpa. (44)
11:51:15.853880 IP _gateway.domain > lab1003-HP-280-G2-MT.40288: 20966 NXDomain* 0/0/0 (44)
11:51:15.855699 IP lab1003-HP-280-G2-MT.40645 > _gateway.domain: 41652+ PTR? 1.1.168.192.in-addr.arpa. (42)
11:51:15.856980 IP _gateway.domain > lab1003-HP-280-G2-MT.40645: 41652 NXDomain* 0/0/0 (42)
11:51:15.857989 IP lab1003-HP-280-G2-MT.51007 > _gateway.domain: 46973+ PTR? 133.1.168.192.in-addr.arpa. (44)
11:51:15.859152 IP _gateway.domain > lab1003-HP-280-G2-MT.51007: 46973 NXDomain* 0/0/0 (44)
11:51:15.860434 IP lab1003-HP-280-G2-MT.55579 > _gateway.domain: 40350+ PTR? 107.1.168.192.in-addr.arpa. (44)
^C
9 packets captured
10 packets received by filter
1 packet dropped by kernel
```

```
lab1003@lab1003-HP-280-G2-MT:~$ sudo tcpdump -v
tcpdump: listening on enp5s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:51:48.291325 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto UDP (17), length 57)
    lab1003-HP-280-G2-MT.54184 > bom07s32-in-f10.1e100.net.443: UDP, length 29
11:51:48.293907 IP (tos 0x0, ttl 64, id 2749, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.37131 > _gateway.domain: 61439+ PTR? 107.1.168.192.in-addr.arpa. (44)
11:51:48.295363 IP (tos 0x0, ttl 63, id 45531, offset 0, flags [DF], proto UDP (17), length 72)
    _gateway.domain > lab1003-HP-280-G2-MT.37131: 61439 NXDomain* 0/0/0 (44)
11:51:48.296599 IP (tos 0x0, ttl 64, id 11676, offset 0, flags [DF], proto UDP (17), length 70)
    lab1003-HP-280-G2-MT.53813 > _gateway.domain: 37981+ PTR? 1.1.168.192.in-addr.arpa. (42)
11:51:48.297803 IP (tos 0x0, ttl 63, id 45532, offset 0, flags [DF], proto UDP (17), length 70)
    _gateway.domain > lab1003-HP-280-G2-MT.53813: 37981 NXDomain* 0/0/0 (42)
11:51:48.319847 IP (tos 0x80, ttl 58, id 0, offset 0, flags [DF], proto UDP (17), length 55)
    bom07s32-in-f10.1e100.net.443 > lab1003-HP-280-G2-MT.54184: UDP, length 27
11:51:48.850813 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.100 tell _gateway, length 46
11:51:48.850879 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.112 tell _gateway, length 46
11:51:48.850957 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.116 tell _gateway, length 46
11:51:48.851042 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.104 tell _gateway, length 46
11:51:48.851494 IP (tos 0x0, ttl 64, id 50385, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.32870 > _gateway.domain: 15790+ PTR? 100.1.168.192.in-addr.arpa. (44)
11:51:48.852960 IP (tos 0x0, ttl 63, id 45538, offset 0, flags [DF], proto UDP (17), length 72)
    _gateway.domain > lab1003-HP-280-G2-MT.32870: 15790 NXDomain* 0/0/0 (44)
11:51:48.854859 IP (tos 0x0, ttl 64, id 27024, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.59607 > _gateway.domain: 60308+ PTR? 112.1.168.192.in-addr.arpa. (44)
11:51:48.856239 IP (tos 0x0, ttl 63, id 45539, offset 0, flags [DF], proto UDP (17), length 72)
    _gateway.domain > lab1003-HP-280-G2-MT.59607: 60308 NXDomain* 0/0/0 (44)
11:51:48.857586 IP (tos 0x0, ttl 64, id 54417, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.42207 > _gateway.domain: 51565+ PTR? 116.1.168.192.in-addr.arpa. (44)
11:51:48.860138 IP (tos 0x0, ttl 64, id 28668, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.46894 > _gateway.domain: 17002+ PTR? 104.1.168.192.in-addr.arpa. (44)
^C
16 packets captured
18 packets received by filter
2 packets dropped by kernel
```

```
lab1003@lab1003-HP-280-G2-MT:~$ sudo tcpdump -vv
tcpdump: listening on enp5s0, link-type EN10MB (Ethernet), capture size 262144 bytes
11:52:20.851630 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.100 tell _gateway, length 46
11:52:20.851779 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.125 tell _gateway, length 46
11:52:20.851887 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 192.168.1.142 tell _gateway, length 46
11:52:20.853252 IP (tos 0x0, ttl 64, id 40225, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.41031 > _gateway.domain: [bad udp cksum 0x8402 -> 0x0b14!] 4493+ PTR? 100.1.168.192.in-addr.arpa. (44)
11:52:20.854720 IP (tos 0x0, ttl 63, id 46573, offset 0, flags [DF], proto UDP (17), length 72)
    _gateway.domain > lab1003-HP-280-G2-MT.41031: [udp sum ok] 4493 NXDomain* q: PTR? 100.1.168.192.in-addr.arpa. 0/0/0 (44)
11:52:20.856512 IP (tos 0x0, ttl 64, id 6385, offset 0, flags [DF], proto UDP (17), length 70)
    lab1003-HP-280-G2-MT.54210 > _gateway.domain: [bad udp cksum 0x8400 -> 0x7642!] 42263+ PTR? 1.1.168.192.in-addr.arpa. (42)
11:52:20.857856 IP (tos 0x0, ttl 63, id 46574, offset 0, flags [DF], proto UDP (17), length 70)
    _gateway.domain > lab1003-HP-280-G2-MT.54210: [udp sum ok] 42263 NXDomain* q: PTR? 1.1.168.192.in-addr.arpa. 0/0/0 (42)
11:52:20.858858 IP (tos 0x0, ttl 64, id 25194, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.36146 > _gateway.domain: [bad udp cksum 0x8402 -> 0x6ba2!] 49678+ PTR? 125.1.168.192.in-addr.arpa. (44)
11:52:20.861423 IP (tos 0x0, ttl 64, id 23503, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.42459 > _gateway.domain: [bad udp cksum 0x8402 -> 0x48ba!] 51792+ PTR? 142.1.168.192.in-addr.arpa. (44)
11:52:20.863992 IP (tos 0x0, ttl 64, id 27140, offset 0, flags [DF], proto UDP (17), length 72)
    lab1003-HP-280-G2-MT.51829 > _gateway.domain: [bad udp cksum 0x8402 -> 0x4e28!] 42051+ PTR? 107.1.168.192.in-addr.arpa. (44)
^C
10 packets captured
13 packets received by filter
3 packets dropped by kernel
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

22) ***curl*** (windows) :

Client URL (cURL, pronounced “curl”) is a command line tool that enables data exchange between a device and a server through a terminal. Using this command line interface (CLI), a user specifies a server URL (the location where they want to send a request) and the data they want to send to that server URL.curl is a command-line tool to transfer data to or from a server, using any of the supported protocols (HTTP, FTP, IMAP, POP3, SCP, SFTP, SMTP, TFTP, TELNET, LDAP, or FILE). curl is powered by Libcurl. This tool is preferred for automation since it is designed to work without user interaction. curl can transfer multiple files at once.

Syntax : curl [options] [URL...]
Parameters :
**-o:** saves the downloaded file on the local machine with the name provided in
**-u:** curl also provides options to download files from user authenticated FTP servers. curl -u {username}:{password} [FTP_URL]

```
C:\Users\lab1003>curl.exe
curl: try 'curl --help' for more information

C:\Users\lab1003>curl --help
Usage: curl [options...] <url>
 -d, --data <data>          HTTP POST data
 -f, --fail                 Fail fast with no output on HTTP errors
 -h, --help <category>      Get help for commands
 -i, --include              Include protocol response headers in the output
 -o, --output <file>        Write to file instead of stdout
 -O, --remote-name          Write output to a file named as the remote file
 -s, --silent               Silent mode
 -T, --upload-file <file>   Transfer local FILE to destination
 -u, --user <user:password> Server user and password
 -A, --user-agent <name>    Send User-Agent <name> to server
 -v, --verbose              Make the operation more talkative
 -V, --version              Show version number and quit

This is not the full help, this menu is stripped into categories.
Use "--help category" to get an overview of all categories.
For all options use the manual or "--help all".

C:\Users\lab1003>curl --version
curl 8.0.1 (Windows) libcurl/8.0.1 Schannel WinIDN
Release-Date: 2023-03-20
Protocols: dict file ftp ftps http https imap imaps pop3 pop3s smtp smtps telnet tftp
Features: AsynchDNS HSTS HTTPS-proxy IDN IPv6 Kerberos Largefile NTLM SPNEGO SSL SSPI threadsafe Unicode UnixSockets
```

```
C:\Users\lab1003>curl --help -d
Usage: curl [options...] <url>
Invalid category provided, here is a list of all categories:

 auth         Different types of authentication methods
 connection   Low level networking operations
 curl         The command line tool itself
 dns          General DNS options
 file         FILE protocol options
 ftp          FTP protocol options
 http         HTTP and HTTPS protocol options
 imap         IMAP protocol options
 misc         Options that don't fit into any other category
 output       Filesystem output
 pop3         POP3 protocol options
 post         HTTP Post specific options
 proxy        All options related to proxies
 scp          SCP protocol options
 sftp         SFTP protocol options
 smtp         SMTP protocol options
 ssh          SSH protocol options
 telnet       TELNET protocol options
 tftp         TFTP protocol options
 tls          All TLS/SSL related options
 upload       All options for uploads
 verbose      Options related to any kind of command line output of curl
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 01*

23) **wget** (Unix/Linux) :

**Wget** is the non-interactive network downloader which is used to download files from the server even when the user has not logged on to the system and it can work in the background without hindering the current process.
GNU wget is a free utility for non-interactive download of files from the Web. wget is non-interactive, meaning that it can work in the background, while the user is not logged on. This allows you to start a retrieval and disconnect from the system, letting wget finish the work. By contrast, most of the Web browsers require constant user's presence, which can be a great hindrance when transferring a lot of data.

Syntax : wget [option] [URL]
Parameters :
**-v / –version :** This is used to display the version of the wget available on your system
**-h / –help :** This is used to print a help message displaying all the possible options of the line command that is available with the wget command line options

```
lab1003@lab1003-HP-280-G2-MT:~$ wget www.facebook.com
--2024-01-25 12:17:26--  http://www.facebook.com/
Resolving www.facebook.com (www.facebook.com)... 31.13.79.35, 2a03:2880:f12f:183:face:b00c:0:25de
Connecting to www.facebook.com (www.facebook.com)|31.13.79.35|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.facebook.com/ [following]
--2024-01-25 12:17:26--  https://www.facebook.com/
Connecting to www.facebook.com (www.facebook.com)|31.13.79.35|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://www.facebook.com/unsupportedbrowser [following]
--2024-01-25 12:17:26--  https://www.facebook.com/unsupportedbrowser
Reusing existing connection to www.facebook.com:443.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/html]
Saving to: ‘index.html.1’

index.html.1            [ <=>                ]  46.11K  --.-KB/s    in 0.02s

2024-01-25 12:17:26 (2.57 MB/s) - ‘index.html.1’ saved [47215]
```

**CONCLUSION :**
**The network assignment on basic networking commands equips you with a foundational skillset for navigating and troubleshooting network issues. We explored essential commands like ping, traceroute, and ipconfig (or ifconfig on macOS/Linux), allowing you to verify connectivity, identify network paths, and configure network settings. With this newfound knowledge, we can effectively diagnose basic network problems and ensure smooth communication within a network infrastructure.**

KUNAL NEMADE
BATCH :S21
ROLL NO. 89

**BASED ON LO1 & LO2**

## NS INSTALLATION

```
lab1003@lab1003-HP-280-G2-MT:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.6 LTS
Release:        18.04
Codename:       bionic
```

```
lab1003@lab1003-HP-280-G2-MT:~$ sudo apt-get install ns2
[sudo] password for lab1003:
Reading package lists... Done
Building dependency tree
Reading state information... Done
ns2 is already the newest version (2.35+dfsg-2.1).
0 upgraded, 0 newly installed, 0 to remove and 53 not upgraded.
```

```
lab1003@lab1003-HP-280-G2-MT:~$ sudo apt-get purge nam
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  nam*
0 upgraded, 0 newly installed, 1 to remove and 53 not upgraded.
After this operation, 683 kB disk space will be freed.
Do you want to continue? [Y/n] Y
(Reading database ... 171305 files and directories currently installed.)
Removing nam (1.15-4) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

KUNAL NEMADE
BATCH :S21
ROLL NO. 89

```
lab1003@lab1003-HP-280-G2-MT:~$ wget   --user  -agent = "Mozilla/5.0 (Windows NT 5.2; rv : 2.0.1) Gecko/20100101 Firefox/4.0.1"   "https:/
hnobytz.com/wp-content/uploads/2015/11/nam_1.14_amd64.zip"
--2024-02-01 11:54:53--  http://=/
Resolving = (=)... failed: Name or service not known.
wget: unable to resolve host address '='
--2024-02-01 11:54:53--  http://mozilla/5.0%20(Windows%20NT%205.2;%20rv%20:%202.0.1)%20Gecko/20100101%20Firefox/4.0.1
Resolving mozilla (mozilla)... failed: Name or service not known.
wget: unable to resolve host address 'mozilla'
--2024-02-01 11:54:53--  https://technobytz.com/wp-content/uploads/2015/11/nam_1.14_amd64.zip
Resolving technobytz.com (technobytz.com)... 18.219.43.235
Connecting to technobytz.com (technobytz.com)|18.219.43.235|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 816308 (797K) [application/zip]
Saving to: 'nam_1.14_amd64.zip'

nam_1.14_amd64.zip                  100%[=============================================================>] 797.18K   102KB/s    in 10s

2024-02-01 11:55:06 (79.8 KB/s) - 'nam_1.14_amd64.zip' saved [816308/816308]

FINISHED --2024-02-01 11:55:06--
Total wall clock time: 13s
Downloaded: 1 files, 797K in 10s (79.8 KB/s)
lab1003@lab1003-HP-280-G2-MT:~$ unzip nam_1.14_amd64.zip
Archive:  nam_1.14_amd64.zip
  inflating: nam_1.14_amd64.deb
lab1003@lab1003-HP-280-G2-MT:~$ Archive: nam_1.14_amd64.zip
Archive:: command not found
lab1003@lab1003-HP-280-G2-MT:~$ unzip nam_1.14_amd64.zip
Archive:  nam_1.14_amd64.zip
replace nam_1.14_amd64.deb? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
  inflating: nam_1.14_amd64.deb
lab1003@lab1003-HP-280-G2-MT:~$ y
y: command not found
```

```
lab1003@lab1003-HP-280-G2-MT:~$ sudo dpkg -i nam_1.14_amd64.deb
[sudo] password for lab1003:
Selecting previously unselected package nam.
(Reading database ... 171298 files and directories currently installed.)
Preparing to unpack nam_1.14_amd64.deb ...
Unpacking nam (1.14) ...
Setting up nam (1.14) ...
lab1003@lab1003-HP-280-G2-MT:~$ nam
Cannot connect to existing nam instance. Starting a new one...
^C
lab1003@lab1003-HP-280-G2-MT:~$ set ns [new Simulator]
lab1003@lab1003-HP-280-G2-MT:~$ nam
Cannot connect to existing nam instance. Starting a new one...
 When configured, ns found the right version of tclsh in /usr/bin/tclsh8.6
but it doesn't seem to be there anymore, so ns will fall back on running the first tclsh in your path. The wrong version of tclsh may break th
e test suites. Reconfigure and rebuild ns if this is a problem.
invalid command name "_o13": invalid command name "_o13"
    while executing
"_o13 renderFrame"
    ("after" script)
```

KUNAL NEMADE
BATCH :S21
ROLL NO. 89

```
lab1003@lab1003-HP-280-G2-MT:~$ set ns [new Simulator]
lab1003@lab1003-HP-280-G2-MT:~$ ns
 When configured, ns found the right version of tclsh in /usr/bin/tclsh8.6
but it doesn't seem to be there anymore, so ns will fall back on running the first tclsh in your path. The wrong version of tclsh may break th
e test suites. Reconfigure and rebuild ns if this is a problem.
% ns at 1 "puts\"Kunal Nemade S21 89\""
warning: using backward compatibility mode
1
% ns at 1.5 "exit"
2
% ns run
ns: puts"Kunal Nemade S21 89": invalid command name "puts"Kunal"
    while executing
"puts"Kunal Nemade S21 89""
```

wget --user -agent = "Mozilla/5.0 (Windows NT 5.2; rv : 2.0.1) Gecko/20100101 Firefox/4.0.1" "https://technobytz.com/wp-content/uploads/2015/11/nam_1.14_amd64.zip"

unzip nam_1.14_amd64.zip
Archive: nam_1.14_amd64.zip
infalting: nam_1.14_amd64.deb

**CONCLUSION :**
**The network assignment on NS-2 equips us with a powerful tool for simulating network behavior.We likely grasped the fundamentals of working with NS-2, a network simulator that allows you to model and test various network scenarios in a controlled environment. This newfound skill is invaluable for designing, troubleshooting, and optimizing network protocols and configurations without the risks associated with real-world deployments.**

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 03*

**AIM** : To implement specific network topologies with respect to TCP. **BASED ON LO3**

## CODE :

```
#Create a simulator object set ns [new Simulator]
#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red
#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf
set np [open out.tr w]
$ns trace-all $np
#Define a 'finish' procedure
proc finish {} {
 global ns nf np
 $ns flush-trace
 #Close the NAM trace file
 close $nf
 #Execute NAM on the trace file
exec nam out.nam &
 exit 0
}
#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 03*

```
set n3 [$ns node]
#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail
#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10
#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5
#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2
#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent
$udp # setting packet
size
$cbr set packet_size_ 1000
#setting bit rate
$cbr set rate_ 1mb
# setting random false means no noise
$cbr set random_ false
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 03*

**#Schedule events for the CBR and FTP agents**

**$ns at 0.1 "$cbr start"**

**$ns at 4.5 "$cbr stop"**

**#Call the finish procedure after 5 seconds of simulation time**

**$ns at 5.0 "finish"**

**#Print CBR packet size and interval**

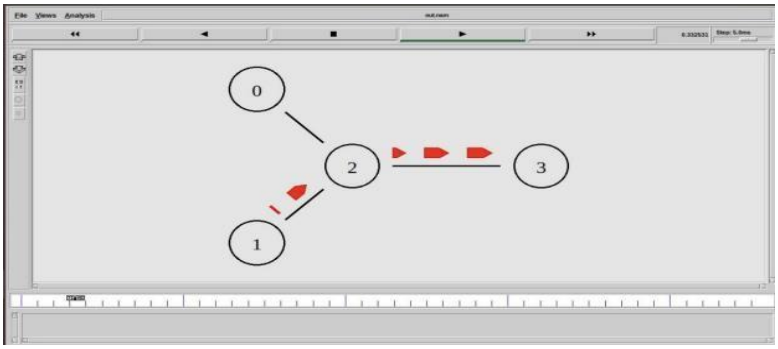
**puts "CBR packet size = [$cbr set packet_size_]"**

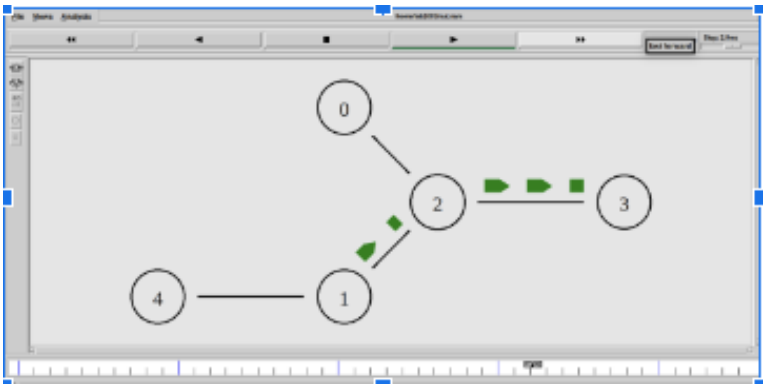
**puts "CBR interval = [$cbr set interval_]"**

**#Run the simulation**

**$ns run**

***OUTPUT :***




**CONCLUSION :**
**Network assignment on implementing a specific network topology with TCP This equips you with a strong understanding of how network structure and protocols interact.We explored how to configure a chosen network topology (like star, bus, or mesh) using network devices and software. Additionally, we delved into the implementation of TCP (Transmission Control Protocol) within this topology, ensuring reliable data transfer between devices. By successfully combining these concepts, we gained valuable practical experience in designing and implementing network communication.**

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 04*

**AIM** : To implement specific network topologies with respect to UDP. **BASED ON LO3**

**CODE** :

```
set ns [new Simulator]
set nr [open out.tr w]
$ns trace-all $nr
set nf [open out.nam w]
$ns namtrace-all $nf
proc finish { } {
global ns nr nf
$ns flush-trace
close $nf
close $nr
exec nam out.nam &
exit 0
}
for { set i 0 } { $i < 4} { incr i 1 } {
set n($i) [$ns node]}
$ns duplex-link $n(0) $n(3) 1Mb 10ms DropTail
$ns duplex-link $n(0) $n(2) 1Mb 10ms DropTail
$ns duplex-link $n(0) $n(1) 1Mb 10ms DropTail
$ns duplex-link $n(1) $n(2) 1Mb 10ms DropTail
$ns duplex-link $n(2) $n(3) 1Mb 10ms DropTail
$ns duplex-link $n(3) $n(1) 1Mb 10ms DropTail
$ns duplex-link-op $n(0) $n(2) orient right
$ns duplex-link-op $n(2) $n(3) orient left-down
$ns duplex-link-op $n(0) $n(3) orient right-down
$ns duplex-link-op $n(3) $n(1) orient left-down
#Setup a TCP connection
set tcp [new Agent/TCP]
$ns attach-agent $n(3) $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n(2) $sink
$ns connect $tcp $sink
$tcp set fid_ 3
#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 04*

```
$ftp set type_ FTP

set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0

set udp1 [new Agent/UDP]
$ns attach-agent $n(1) $udp1

set udp2 [new Agent/UDP]
$ns attach-agent $n(2) $udp2

set udp3 [new Agent/UDP]
$ns attach-agent $n(3) $udp3

set udp4 [new Agent/UDP]
$ns attach-agent $n(1) $udp4

set udp5 [new Agent/UDP]
$ns attach-agent $n(0) $udp5

set null0 [new Agent/Null]
$ns attach-agent $n(0) $null0


set null1 [new Agent/Null]
$ns attach-agent $n(1) $null1


set null2 [new Agent/Null]
$ns attach-agent $n(2) $null2


set null3 [new Agent/Null]
$ns attach-agent $n(3) $null3

set null4 [new Agent/Null]
$ns attach-agent $n(3) $null4

set null5 [new Agent/Null]
$ns attach-agent $n(2) $null5
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 04*

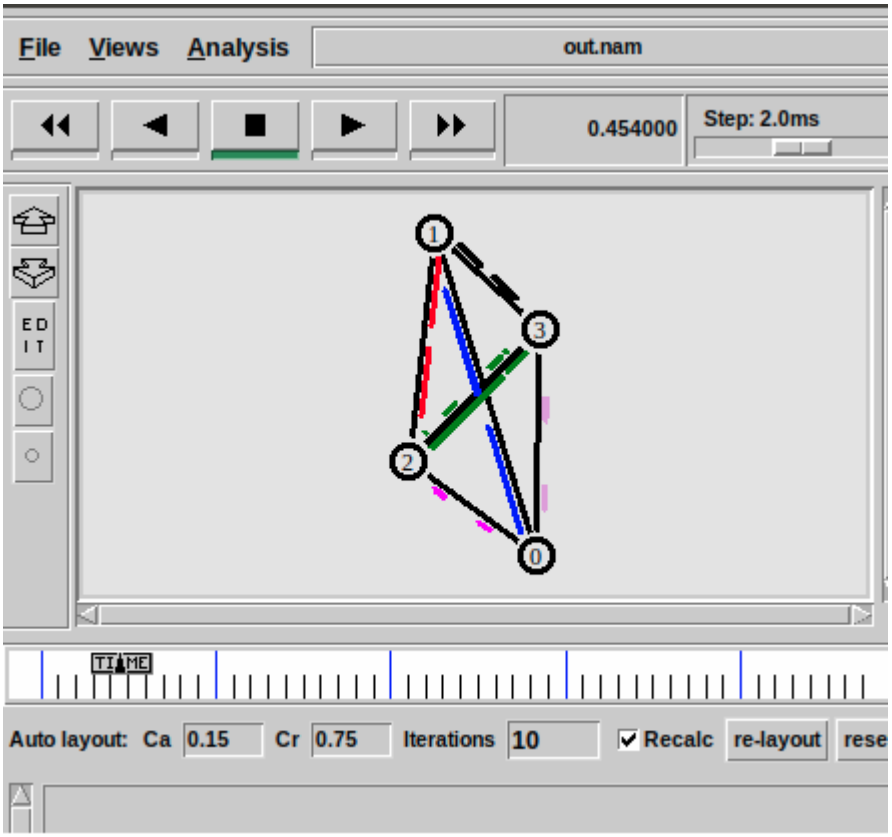
```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0

set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 500
$cbr1 set interval_ 0.005
$cbr1 attach-agent $udp1


set cbr2 [new Application/Traffic/CBR]
$cbr2 set packetSize_ 200
$cbr2 set interval_ 0.005
$cbr2 attach-agent $udp2

set cbr3 [new Application/Traffic/CBR]
$cbr3 set packetSize_ 200
$cbr3 set interval_ 0.005
$cbr3 attach-agent $udp3

set cbr4 [new Application/Traffic/CBR]
$cbr4 set packetSize_ 500
$cbr4 set interval_ 0.005
$cbr4 attach-agent $udp4

set cbr5 [new Application/Traffic/CBR]
$cbr5 set packetSize_ 200
$cbr5 set interval_ 0.005
$cbr5 attach-agent $udp5


$ns connect $udp0 $null1
$ns connect $udp1 $null2
$ns connect $udp2 $null3
$ns connect $udp3 $null0
$ns connect $udp4 $null4
$ns connect $udp5 $null5
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 04*

```
$udp0 set fid_ 1
$udp1 set fid_ 2
$udp2 set fid_ 3
$udp3 set fid_ 6
$udp4 set fid_ 4
$udp5 set fid_ 5
$ns color 4 Black
$ns color 5 Magenta
$ns color 1 Blue
$ns color 2 Red
$ns color 3 Green
$ns color 6 Plum
$ns at 0.1 "$cbr1 start"
$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr2 start"
$ns at 0.1 "$cbr3 start"
$ns at 0.1 "$cbr4 start"
$ns at 0.1 "$cbr5 start"
$ns at 3.0 "$cbr4 stop"
$ns at 3.0 "$cbr5 stop"
$ns at 3.0 "$cbr1 stop"
$ns at 3.0 "$cbr0 stop"
$ns at 3.0 "$cbr2 stop"
$ns at 3.0 "$cbr3 stop"
$ns at 0.2 "$ftp start"
$ns at 4.5 "$ftp stop"
$ns at 45 "finish"
$ns run
```

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 04*

## **OUTPUT** :




**CONCLUSION :**

**The network assignment on implementing a specific network topology with UDP highlights understanding of the trade-offs between different protocols in network design.WE explored configuring a chosen network topology (like star, bus, or mesh) and implemented UDP (User Datagram Protocol) within it. UDP prioritizes speed over reliability, making it ideal for applications where real-time data delivery is crucial, even if some packets might be lost. By successfully implementing UDP in a specific topology, We gained valuable experience in designing networks that prioritize speed and efficiency. Keep practicing to explore different network topologies and how they interact with UDP. This will help you choose the right protocol and network structure for various communication needs, ensuring optimal performance in your network designs.**

KUNAL NEMADE
BATCH : S21
ROLL NO : 89
COMPUTER NETWORK ASSIGNMENT NO :. 6

# NETWOTK SIMULATION WITH DISTANCE VECTOR ROUTING (DVR) PROTOCOL **BASED ON LO4**

**Aim:**
To simulate and study the Distance Vector Routing Algorithm using simulation using NS2.

**Theory:**

Distant vector routing protocol used to calculate a path. A distance-vector protocol calculates the distance and direction of the vector of the next hop from the information obtained by the neighboring router. The distance vector routing algorithm works by having each router maintain a routing table, giving the best-known distance from source to destination and which route is used to get there.

These tables are updated by exchanging the information with the neighbor having a direct link. Tables contain one entry for each route, this entry contains two parts, the preferred outgoing line used to reach the destination or an estimate of the time or distance to that destination.

The metric used can be the number of hops required to reach from source to destination. Time delay in milliseconds, the router can measure it with a special echo signal which the receiver can timestamp and send as soon as possible.

The router exchanges the network topology information periodically with its neighboring node and updates the information in the routing table. The cost of reaching the destination is estimated based on the metric, and an optimal path is obtained to send data packets from source to destination.It is necessary to keep track of the topology and inform neighboring devices if any changes occur in the topology.

**Network Information:**
Every node in the network should have information about its neighboring node. Each node in the network is designed to share information with all the nodes in the network.

**Routing Pattern:**
In DVR the data shared by the nodes are transmitted only to that node that is linked directly to one or more nodes in the network.

**Data sharing:**
The nodes share the information with the neighboring node from time to time as there is a change in network topology.

### The Distance vector algorithm is iterative, asynchronous and distributed.

**Distributed:** It is distributed in that each node receives information from one or more of its directly attached neighbors, performs calculation and then distributes the result back to its neighbors.
**Iterative:** It is iterative in that its process continues until no more information is available to be exchanged between neighbors.
**Asynchronous:** It does not require that all of its nodes operate in the lock step with each other.

The Distance vector algorithm is a dynamic algorithm. It is mainly used in ARPANET, and RIP. Each router maintains a distance table known as **Vector**.

**Knowledge about the whole network:** Each router shares its knowledge through the entire network. The Router sends its collected knowledge about the network to its neighbors.

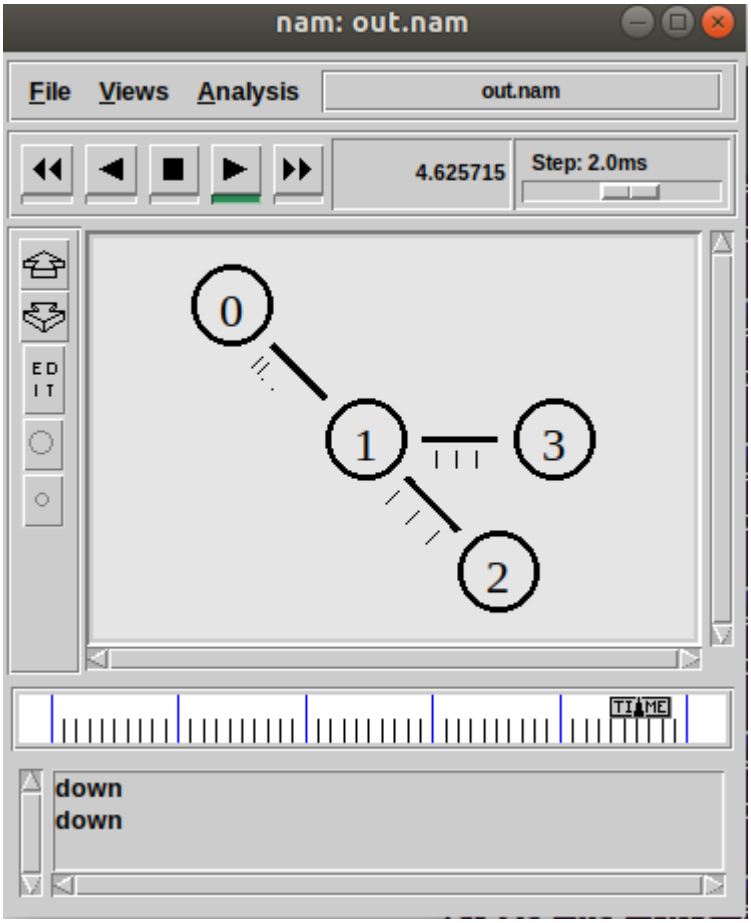
**Routing only to neighbors:** The router sends its knowledge about the network to only those routers which have direct links. The router sends whatever it has about the network through the ports. The information is received by the router and uses the information to update its own routing table.

**Information sharing at regular intervals:** Within 30 seconds, the router sends the information to the neighboring routers.

**CODE :**

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tr [open out.tr w]
$ns trace-all $tr
proc finish {} {
 global nf ns tr
 $ns flush-trace
 close $tr
 exec nam out.nam &
 exit 0
}
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
```

```
$ns duplex-link $n0 $n1 10Mb 10ms DropTail
$ns duplex-link $n1 $n3 10Mb 10ms DropTail
$ns duplex-link $n2 $n1 10Mb 10ms DropTail
$ns duplex-link-op $n0 $n1 orient right-down
$ns duplex-link-op $n1 $n3 orient right
$ns duplex-link-op $n2 $n1 orient left-up


set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set ftp [new Application/FTP]
$ftp attach-agent $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
set udp [new Agent/UDP]
$ns attach-agent $n2 $udp
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $tcp $sink
$ns connect $udp $null
$ns rtmodel-at 1.0 down $n1 $n3
$ns rtmodel-at 2.0 up $n1 $n3
$ns rtproto DV
$ns at 0.0 "$ftp start"
$ns at 0.0 "$cbr start"
$ns at 5.0 "finish"

$ns run
```




**CONCLUSION :**

**The network assignment on routing protocols equips us with a fundamental understanding of how data finds its way across complex networks.We explored different routing protocols, like RIP or OSPF, which allow network devices to communicate and dynamically determine the most efficient path to send data packets. By understanding routing protocols, we can now delve deeper into how networks function and ensure smooth and efficient data flow within various network configurations.**

KUNAL NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO :. 6

# NETWOTK SIMULATION WITH LINK STATE ROUTING PROTOCOL **BASED ON LO4**

**Aim:**
To simulate and study the Link State Routing Algorithm using simulation using NS2.

**Theory:**
In link state routing, each router shares its knowledge of its neighborhood with every otherrouter in the internet work. link-state routing uses link-state routers to exchange messages that allow each router to learn the entire network topology. Based on this learned topology, each router is then able to compute its routing table by using the shortest path computation.Link state routing is a technique in which each router shares the knowledge of its neighborhood with every other router i.e. the internet work. The three keys to understand the link state routing algorithm.

(i) Knowledge about Neighborhood: Instead of sending its entire routing table
a router sendsinfo about its neighborhood only.
(ii) To all Routers: each router sends this information to
every other routeron the internet work not just to its neighbor .It does so by a process called flooding.
(iii)Information sharing
when there is a change: Each router sends out information about the neighbors when there is
change.

**Link state routing has two phase:**

1. **Reliable Flooding: Initial state**– Each node knows the cost of its neighbors. Final state- Each node knows the entire graph.
2. **Route Calculation**: Each node uses Dijkstra' s algorithm on the graph to calculate the optimal routes to all nodes. The link state routing algorithm is also known as Dijkstra's algorithm which is used to find the shortest path from one node to every other node in the network.

## Features of Link State Routing Protocols

- **Link State Packet:** A small packet that contains routing information.
- **Link-State Database:** A collection of information gathered from the link-state packet.
- **Shortest Path First Algorithm (Dijkstra algorithm):** A calculation performed on the database results in the shortest path
- **Routing Table:** A list of known paths and interfaces.

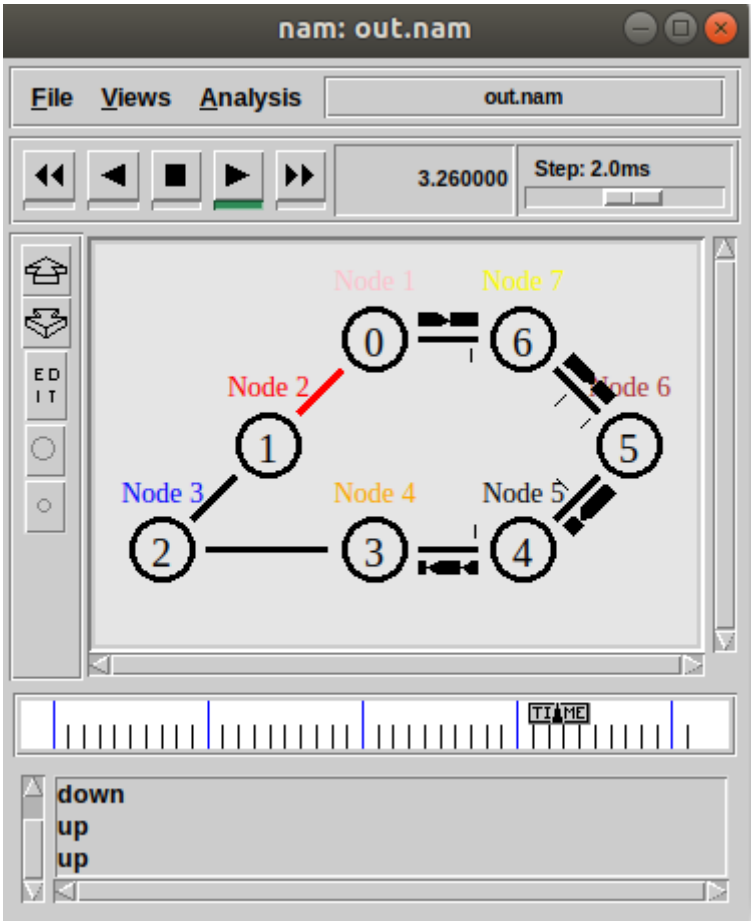
**CODE :**

```
set ns [new Simulator]
$ns rtproto LS
set node1 [$ns node]
set node2 [$ns node]
set node3 [$ns node]
set node4 [$ns node]
set node5 [$ns node]
set node6 [$ns node]
set node7 [$ns node]

set tf [open out.tr w]
$ns trace-all $tf
set nf [open out.nam w]
$ns namtrace-all $nf

$node1 label "Node 1"
$node2 label "Node 2"
$node3 label "Node 3"
$node4 label "Node 4"
$node5 label "Node 5"
$node6 label "Node 6"
$node7 label "Node 7"

$node1 label-color pink
$node2 label-color red
$node3 label-color blue
$node4 label-color orange
$node5 label-color black
$node6 label-color brown
$node7 label-color yellow

$ns duplex-link $node1 $node2 1.5Mb 10ms DropTail
$ns duplex-link $node2 $node3 1.5Mb 10ms DropTail
$ns duplex-link $node3 $node4 1.5Mb 10ms DropTail
$ns duplex-link $node4 $node5 1.5Mb 10ms DropTail
$ns duplex-link $node5 $node6 1.5Mb 10ms DropTail
$ns duplex-link $node6 $node7 1.5Mb 10ms DropTail
$ns duplex-link $node7 $node1 1.5Mb 10ms DropTail

$ns duplex-link-op $node1 $node2 orient left-down
$ns duplex-link-op $node2 $node3 orient left-down
$ns duplex-link-op $node3 $node4 orient right
$ns duplex-link-op $node4 $node5 orient right
$ns duplex-link-op $node5 $node6 orient right-up
$ns duplex-link-op $node6 $node7 orient left-up
$ns duplex-link-op $node7 $node1 orient left
```

```
set tcp2 [new Agent/TCP]
$ns attach-agent $node1 $tcp2
set sink2 [new Agent/TCPSink]
$ns attach-agent $node4 $sink2
$ns connect $tcp2 $sink2

set traffic_ftp2 [new Application/FTP]
$traffic_ftp2 attach-agent $tcp2

proc finish {} {
 global ns nf
 $ns flush-trace
 close $nf
 exec nam out.nam &
 exit 0
}

$ns at 0.5 "$traffic_ftp2 start"
$ns rtmodel-at 1.0 down $node1 $node2
$ns rtmodel-at 2.0 up $node2 $node3
$ns at 3.0 "$traffic_ftp2 start"
$ns at 4.0 "$traffic_ftp2 stop"
$ns at 5.0 "finish"

$ns run
```

**CONCLUSION :**

**The network assignment on routing protocols equips us with a fundamental understanding of how data finds its way across complex networks.We explored different routing protocols, like RIP or OSPF, which allow network devices to communicate and dynamically determine the most efficient path to send data packets. By understanding routing protocols, we can now delve deeper into how networks function and ensure smooth and efficient data flow within various network configurations.**



KUNAL NEMADE
S2 BATCH : S21
ROLL NO. 89

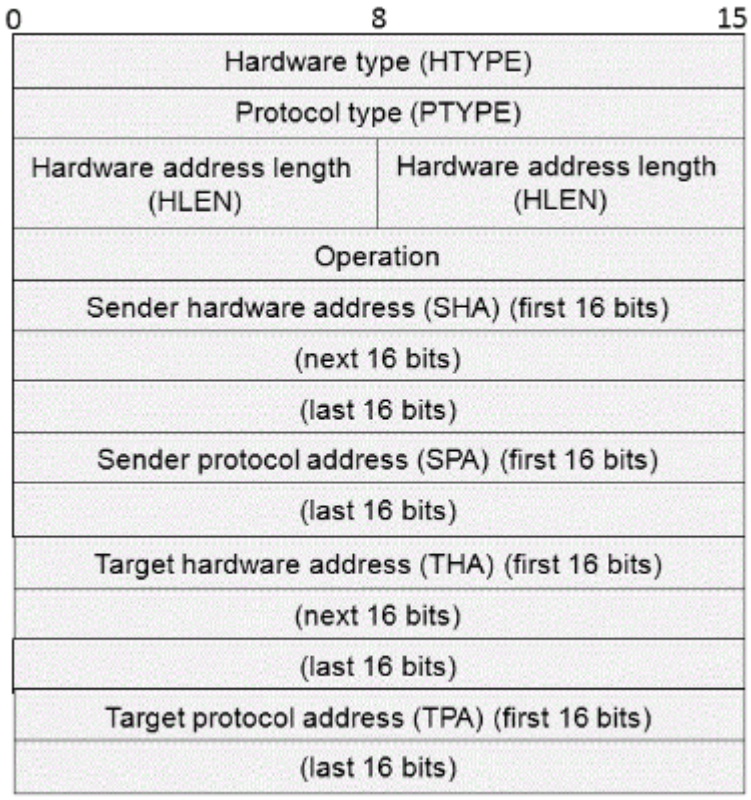
# COMPUTER NETWORKING LAB

**BASED ON LO5**

## WIRESHARK

**AIM : Establish a WireShark Connection and analyse the protocols and its headers :**

### 1. ADDRESS RESOLUTION PROTOCOL (ARP) :

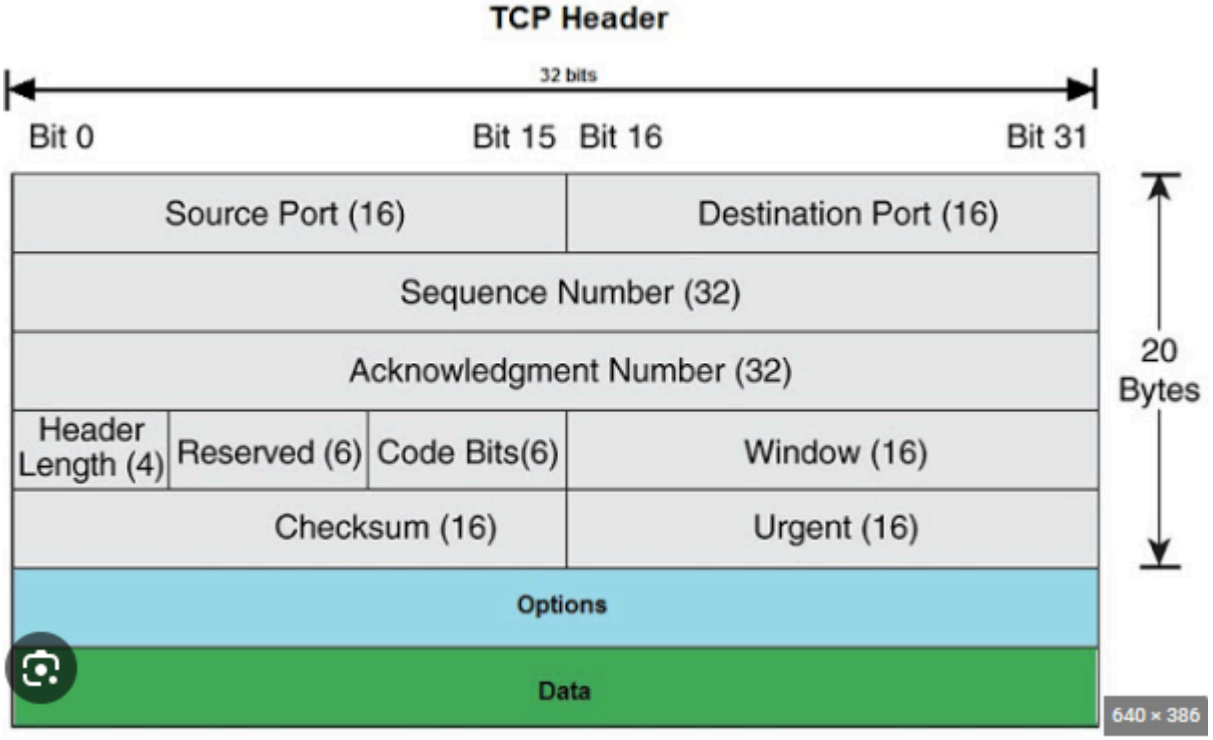



```
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: HewlettPacka_49:6c:fc (f4:39:09:49:6c:fc)
    Sender IP address: 192.168.0.160
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.0.224
```

# KUNAL NEMADE
# S2 BATCH : S21
# ROLL NO. 89
# COMPUTER NETWORKING LAB

HARWARE TYPE : ETHERNET (1)
PROTOCOL TYPE : IVP4(0X0800)
TARGET IP ADDRESS : 192.168.0.224
SENDER  IP ADDRESS : 192.168.0.160

## 2. TRANMISSION CONTROL PROTOCOL (TCP)

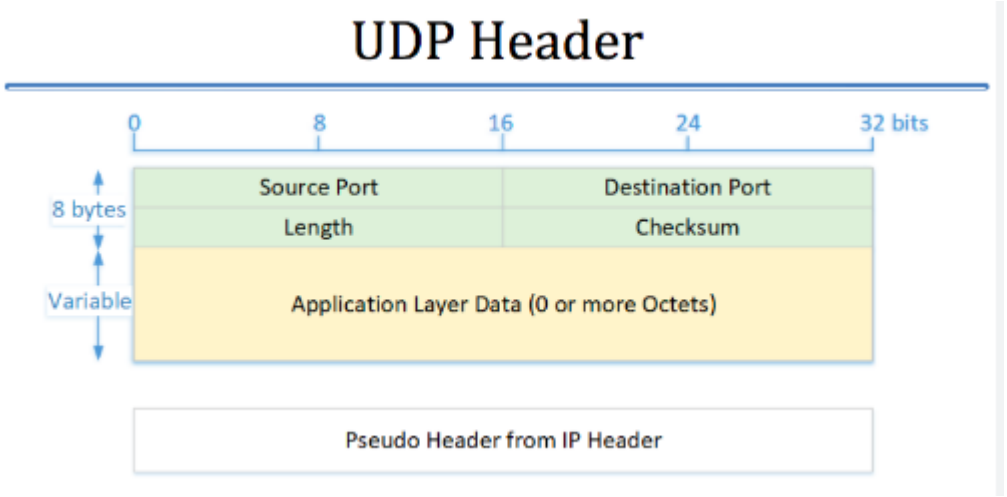



```
Transmission Control Protocol, Src Port: 7680, Dst Port: 51453, Seq: 661672251, Ack: 17133,
    Source Port: 7680
    Destination Port: 51453
    [Stream index: 726]
  > [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 1460]
    Sequence Number: 661672251    (relative sequence number)
    Sequence Number (raw): 494373104
    [Next Sequence Number: 661673711    (relative sequence number)]
    Acknowledgment Number: 17133    (relative ack number)
    Acknowledgment number (raw): 4133080660
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
    Window: 8209
```

# KUNAL NEMADE
# S2 BATCH : S21
# ROLL NO. 89
# COMPUTER NETWORKING LAB

SOURCE PORT :7680
DESTINATION PORT : 51453
SEQUENCE NUMBER : 661672251
WINDOW : 8209

### 3. USER DATAGRAM PROTOCOL (UDP) :

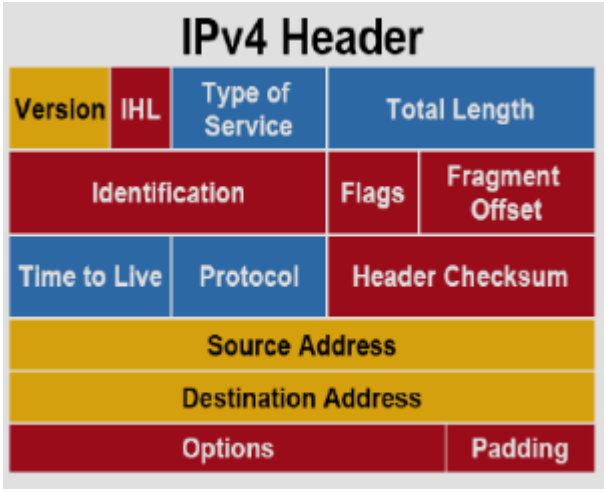



```
User Datagram Protocol, Src Port: 53, Dst Port: 64257
    Source Port: 53
    Destination Port: 64257
    Length: 141
    Checksum: 0xcc87 [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
  > [Timestamps]
    UDP payload (133 bytes)
```

Source Port : 53
Destination Port : 64257
Length : 141
Checksum : 0xcc87  [unverified]

# KUNAL NEMADE
# S2 BATCH : S21
# ROLL NO. 89
# COMPUTER NETWORKING LAB

## 4. INTERNET PROTOCOL VERSION 4



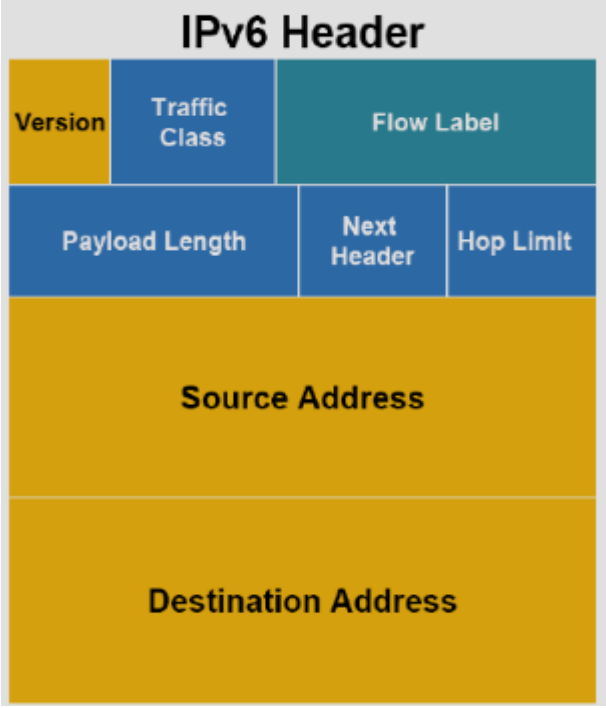

```
Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 334
    Identification: 0xdc21 (56353)
  > 000. .... = Flags: 0x0
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 128
    Protocol: UDP (17)
    Header Checksum: 0x5d7e [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 0.0.0.0
    Destination Address: 255.255.255.255
```

Internet Protocol Version 4,
Src: 0.0.0.0,
Dst: 255.255.255.255 0100
Version: 4.... 0101
Header Length: 20 bytes (5)
Identification: 0xdc21 (56353)
Time to Live: 128
Protocol: UDP (17)

# KUNAL NEMADE
# S2 BATCH : S21
# ROLL NO. 89
# COMPUTER NETWORKING LAB

Header Checksum: 0x5d7e [validation disabled]
[Header checksum status: Unverified]
Source Address: 0.0.0.0
Destination Address: 255.255.255.255

## 5. INTERNET PROTOCOL VERSION 6




```
Internet Protocol Version 6, Src: fe80::f6c1:63f8:95aa:8f93, Dst: ff02::1:2
    0110 .... = Version: 6
  > .... 0000 0000 .... .... .... .... .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
    .... 0010 1111 1000 1110 0000 = Flow Label: 0x2f8e0
    Payload Length: 103
    Next Header: UDP (17)
    Hop Limit: 1
    Source Address: fe80::f6c1:63f8:95aa:8f93
    Destination Address: ff02::1:2
```

# KUNAL NEMADE
# S2 BATCH : S21
# ROLL NO. 89
# COMPUTER NETWORKING LAB

Internet Protocol Version 6,

Src: fe80::f6c1:63f8:95aa:8f93,

Dst: ff02::1:2 0110 .... = Version: 6

0000 0000

Traffic Class: 0x00 (DSCP: CSO, ECN: Not-ECT) 0010 1111 1000 1110 0000
Flow Label: 0x2f8e0

Payload Length: 103

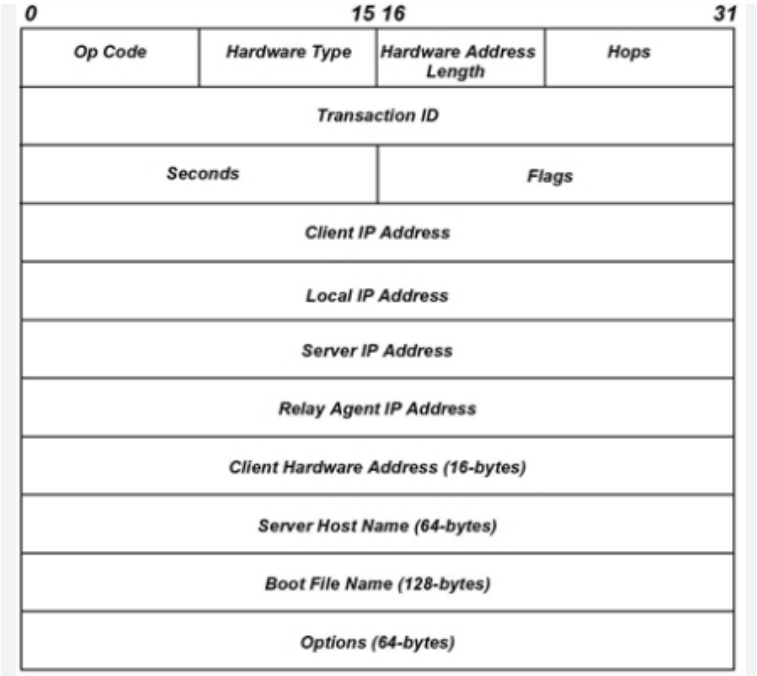
Next Header: UDP (17)

Hop Limit: 1

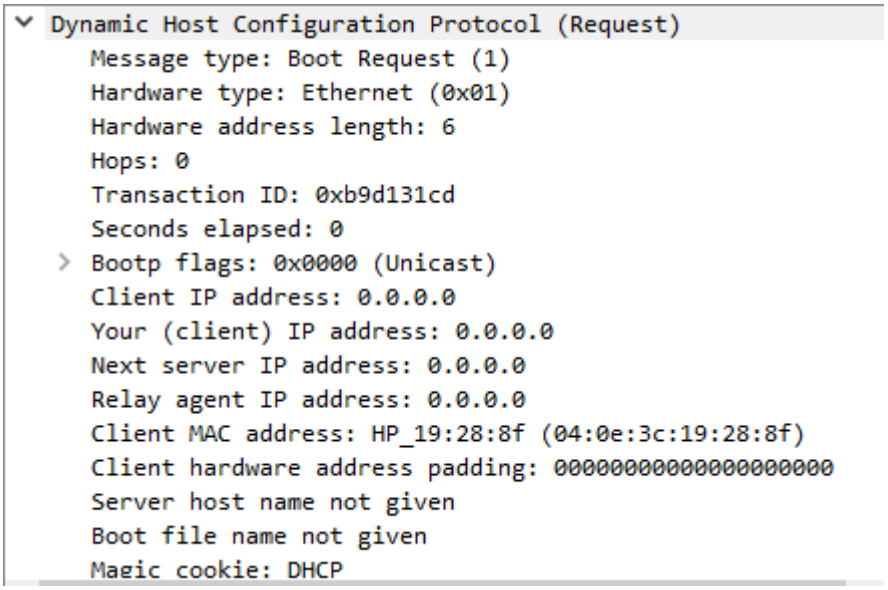
Source Address: fe80::f6c1:63f8:95aa:8f93

Destination Address: ff02::1:2

## 6. DOMAIN HOST CONFIGURATION PROTOCOL (DHCP)



# KUNAL NEMADE
# S2 BATCH : S21
# ROLL NO. 89
# COMPUTER NETWORKING LAB




Dynamic Host Configuration Protocol (Request)
Message type: Boot Request
(1) Hardware type: Ethernet (0x01)
Hardware address length: 6
Hops: 0
Transaction ID: 0xb9d131cd Seconds elapsed: 0
> Bootp flags: 0x0000 (Unicast) Client IP address: 0.0.0.0
Your (client) IP address: 0.0.0.0 Next server IP address: 0.0.0.0
Relay agent IP address: 0.0.0.0
Client MAC address: HP 19:28:8f (04:0e: 3c:19:28:8f)
Client hardware address padding: 00000000000000000000

**CONCLUSION :**
**The network assignment on installing Wireshark equips you with a valuable tool for network analysis and troubleshooting. Wireshark is a powerful packet sniffer that allows you to capture and inspect network traffic data flowing across your network. By successfully installing Wireshark**

KUNAL
NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO:5

**BASED ON LO6**

# SOCKET PROGRAMMING USING TCP/IP

**Establish a Socket Connection**

To connect to another machine we need a socket connection. A socket connection means the two machines have information about each other's network location (IP Address) and TCP port. The java.net.Socket class represents a Socket.

**STEP 1 :** First write the Java Program for Client application

```
import java.io.*;
import java.net.*;

public class Client {
   public static void main(String[] args) {
      try {
         Socket socket = new Socket("localhost", 5000);
         System.out.println("Connected to server.");

         BufferedReader reader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
         PrintWriter writer = new PrintWriter(socket.getOutputStream(), true);

         BufferedReader consoleReader = new BufferedReader(new
InputStreamReader(System.in));

         String inputLine, outputLine;
         while (true) {
            System.out.print("Client: ");
            outputLine = consoleReader.readLine();
            writer.println(outputLine);
```

KUNAL
NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO:5

```
            if (outputLine.equalsIgnoreCase("bye")) break;

            inputLine = reader.readLine();
            System.out.println("Server: " + inputLine}

          writer.close();
          reader.close();
          socket.close()
          ;
       } catch (IOException e) {
          e.printStackTrace();
       }
    }
}
```

KUNAL
NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO:5

**STEP 2 :** First write the Java Program for Server application

```
import java.io.*;
import java.net.*;

public class Server {
   public static void main(String[] args) {
      try {
         ServerSocket serverSocket = new ServerSocket(5000);
         System.out.println("Server started, waiting for client...");

         Socket clientSocket = serverSocket.accept();
         System.out.println("Client connected: " + clientSocket);

         BufferedReader reader = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
         PrintWriter writer = new PrintWriter(clientSocket.getOutputStream(), true);

         BufferedReader consoleReader = new BufferedReader(new
InputStreamReader(System.in));

         String inputLine, outputLine;
         while ((inputLine = reader.readLine()) != null) {
            System.out.println("Client: " + inputLine);


            System.out.print("Server: ");
            outputLine = consoleReader.readLine();
            writer.println(outputLine);
            if (outputLine.equalsIgnoreCase("bye")) break;
         }

         writer.close();
         reader.close();
         clientSocket.close();
```

KUNAL
NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO:5

```
            serverSocket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

KUNAL
NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO 5

**STEP 3 :** First run the Server application

```
lab1003@lab1003-HP-280-G2-MT:~$ cd Downloads
lab1003@lab1003-HP-280-G2-MT:~/Downloads$ java Server.java
Server started
Waiting for a client ...
Client accepted
Hello ! Its connected
```

```
Server:
Client: Hello ! Its connected
Server: Yup ! Its connected
```

**STEP 4 :** First run the Client application

```
lab1003@lab1003-HP-280-G2-MT:~/Downloads$ java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
Hello ! Its connected
```

```
Client: Hello ! Its connected
Server: Yup ! Its connected
Client:
```

z

KUNAL
NEMADE
BATCH :S21
ROLL NO. 89
COMPUTER NETWORK ASSIGNMENT NO 5

**STEP 5 :** After successfully establishing the connection, write the message in the Client terminal/window.

```
lab1003@lab1003-HP-280-G2-MT:~/Downloads$ java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
Connected
Hello ! Its connected
```

```
Server:
Client: Hello ! Its connected
Server: Yup ! Its connected
```

**STEP 6 :** The message will be simultaneosly received and be showed on the Server Window/Terminal

```
lab1003@lab1003-HP-280-G2-MT:~$ cd Downloads
lab1003@lab1003-HP-280-G2-MT:~/Downloads$ java Server.java
Server started
Waiting for a client ...
Client accepted
Hello ! Its connected
```

```
Client: Hello ! Its connected
Server: Yup ! Its connected
Client:
```

**CONCLUSION :**
**The network assignment on socket programming using TCP equips us with a foundational skill for building network applications.We likely explored the concepts of sockets, which act as endpoints for communication, and delved into the TCP protocol, ensuring reliable data transfer between programs on different machines.**

# *KUNAL NEMADE*
# *ROLL NO :. 89*
# *NL LAB SE/S2/89/ASSIGNMENT NO :. 09*

**AIM** : To implement Socket programming using UDP. **BASED ON ,LO5**

**THEORY** :

Socket programming enables communication between programs over a network by establishing connections, sending, and receiving data. It provides a means for applications to interact with each other, whether they are running on the same device or on different devices connected over a network. This communication can occur using different network protocols such as TCP or UDP, allowing for various types of interactions, including client-server communication, peer-to- peer communication, and data exchange between applications. Socket programming is essential for building networked applications, enabling them to communicate, share information, and coordinate actions across distributed systems

ONE WAY:
CLIENT SIDE:

```java
import java.net.*;

public class roclient {
    public static void main(String[] args) {
        try {
            DatagramSocket clientSocket = new DatagramSocket();
            InetAddress IPAddress = InetAddress.getByName("localhost");
            byte[] sendData;
            byte[] receiveData = new byte[1024];
            String sentence = "Hello Server!";
            sendData = sentence.getBytes();

            // Send data to server
            DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress, 9876);
            clientSocket.send(sendPacket);

            // Receive response from server
            DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
            clientSocket.receive(receivePacket);
            String modifiedSentence = new String(receivePacket.getData());
            System.out.println("From Server: " + modifiedSentence);

            clientSocket.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

**OUTPUT:**

***KUNAL NEMADE***

***ROLL NO :. 89***

***NL LAB SE/S2/89/ASSIGNMENT NO :. 09***

```
lab1003@lab1003-HP-280-G2-MT:~/Documents$ javac roclient.java
lab1003@lab1003-HP-280-G2-MT:~/Documents$ java roclient.java
error: class found on application class path: roclient
lab1003@lab1003-HP-280-G2-MT:~/Documents$ java roclient
From Server: HELLO SERVER!
```

**SERVER SIDE:**

```
import java.net.*;

public class roserver {
    public static void main(String[] args) {
        try {
            DatagramSocket serverSocket = new DatagramSocket(9876);
            byte[] receiveData = new byte[1024];
            byte[] sendData = new byte[1024];

            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
                serverSocket.receive(receivePacket);
                String sentence = new String(receivePacket.getData());
                InetAddress IPAddress = receivePacket.getAddress();
                int port = receivePacket.getPort();
                System.out.println("Received: " + sentence);

                // Modify the received data (if needed)
                String modifiedSentence = sentence.toUpperCase();
                sendData = modifiedSentence.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress, port);
                serverSocket.send(sendPacket);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

**OUTPUT:**

```
^Clab1003@lab1003-HP-280-G2-MT:~/Documents$ javac roserver.java
lab1003@lab1003-HP-280-G2-MT:~/Documents$ java roserver.java
error: class found on application class path: roserver
lab1003@lab1003-HP-280-G2-MT:~/Documents$ java roserver
Received: Hello Server!
^Clab1003@lab1003-HP-280-G2-MT:~/Documents$
```

**TWO WAYS:**

**SERVER SIDE:**

# KUNAL NEMADE
# ROLL NO :. 89
# NL LAB SE/S2/89/ASSIGNMENT NO :. 09

```java
import java.io.*;
import java.net.*;

public class rohserver {
    public static void main(String[] args) {
        try (DatagramSocket serverSocket = new DatagramSocket(9876)) {
            System.out.println("Server started and listening...");

            byte[] receiveData = new byte[1024];
            byte[] sendData = new byte[1024];

            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
                serverSocket.receive(receivePacket);
                String clientMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());
                System.out.println("Client: " + clientMessage);

                InetAddress clientAddress = receivePacket.getAddress();
                int clientPort = receivePacket.getPort();

                // Server sends response
                BufferedReader consoleReader = new BufferedReader(new InputStreamReader(System.in));
                System.out.print("Server: ");
                String response = consoleReader.readLine();
                sendData = response.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, clientAddress, clientPort);
                serverSocket.send(sendPacket);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

## OUTPUT:

```
^Clab1003@lab1003-HP-280-G2-MT:~/Documents$ java rohserver.java
Server started and listening...
Client: hello rohan
Server: hello kunal
Client: hello mortal
Server: hello pratik
```

## CLIENT SIDE:

```java
import java.io.*;
import java.net.*;

public class rohclient {
    public static void main(String[] args) {
        try (DatagramSocket clientSocket = new DatagramSocket()) {
            InetAddress serverAddress = InetAddress.getByName("localhost");
            byte[] sendData = new byte[1024];
            byte[] receiveData = new byte[1024];

            BufferedReader consoleReader = new BufferedReader(new InputStreamReader(System.in));

            while (true) {
                System.out.print("Client: ");
                String message = consoleReader.readLine();
                sendData = message.getBytes();
                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, serverAddress, 9876);
                clientSocket.send(sendPacket);

                DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
                clientSocket.receive(receivePacket);
                String serverMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());
                System.out.println("Server: " + serverMessage);

                if (message.equalsIgnoreCase("bye") || serverMessage.equalsIgnoreCase("bye")) {
                    break;
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

*KUNAL NEMADE*
*ROLL NO :. 89*
*NL LAB SE/S2/89/ASSIGNMENT NO :. 09*

**OUTPUT:**

```
lab1003@lab1003-HP-280-G2-MT:~/Documents$ java rohclient.java
Client: hello rohan
Server: hello kunal
Client: hello mortal
Server: hello pratik
Client:
```

**CONCLUSION :**
**The network assignment on socket programming using UDP equips us with the skills to build applications that prioritize speed over guaranteed delivery. We likely explored creating sockets, the endpoints for communication, and delved into the UDP (User Datagram Protocol) protocol. UDP prioritizes speed by sending data packets without error checking or guaranteed delivery, making it ideal for real-time applications where immediate response is crucial, even if some data might be lost.**

CNND

WRITTEN ASSIGNMENT - I

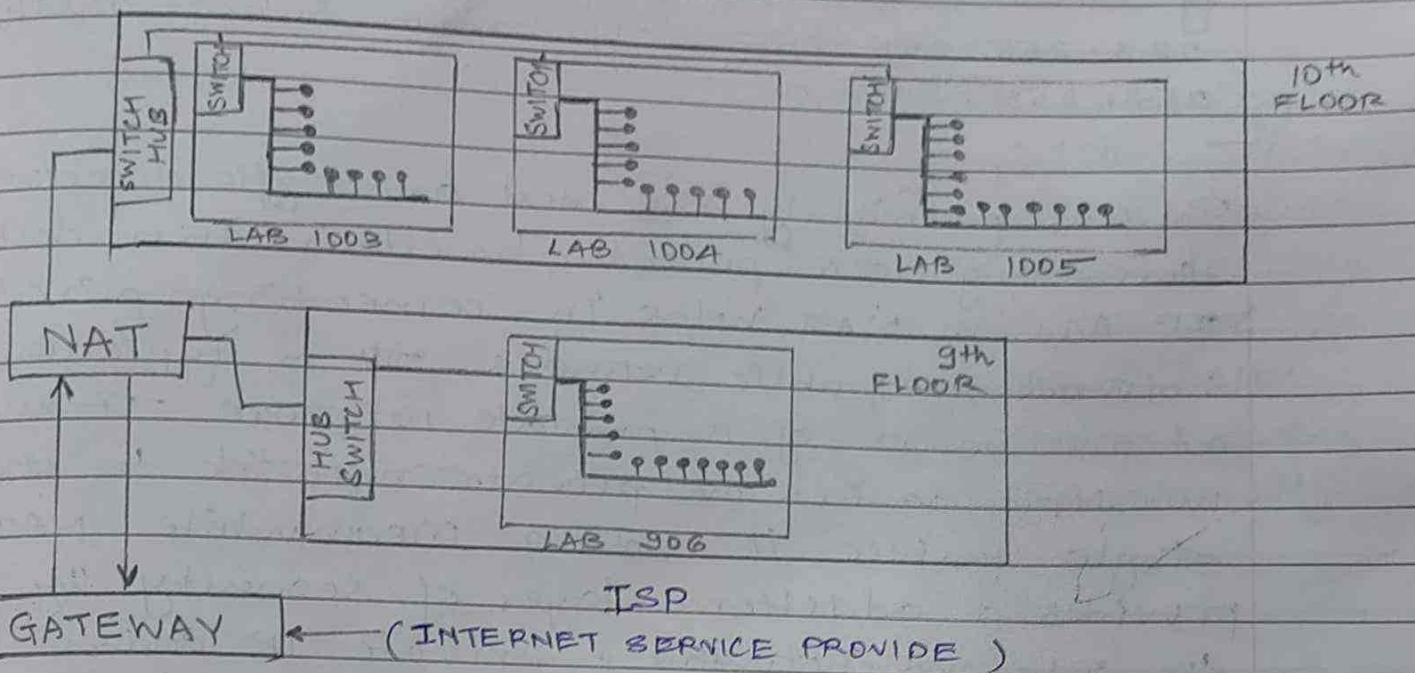
07/03/2024

DOMS | Page No.

Date / /

1) Design and configure any organisation network.

Ans The physical network span across 2 floors consisting of 200 computers in 4 labs. Three labs are on the 10th floor and one labs on the 9th floor. There are around 50 computers in lab 1003, 1004 and 1005 and 906 on the 9th floor. All the labs will be connected with the formation in Tree Topology via Hub while the individual computers in the lab will be connected in Bus Topology to a switch. The hubs of the respective labs will be connected to a gateway. Network Address Translation (NAT) will ensure to map private IP Address to public address.




The layout of the lab resemble tree topology. Each node in a particular lab is connected to hub/switch in the form of bus topology due to less cost, low dependance, and easy access to hub. The hub of all the labs are then connected to the hub of

Floors
lab which in turn are connected to NAT and then they are connected to Gateway where Internet Service Provider help with IP addressing.

Computers in the lab are connected with the help Ethernet cable with the help of RJ45 connector. The entire network organisation is connected to the outside world using Local Area Network (LAN). Number of nodes in total are 200. Thus we will be Class C Addressing of Classfull IP Addressing. It has 16 bit of Network Id and 8 bit of host id. The starting of class C are 110 followed by 13 bits of Network Id. Range of Class C Addressing goes from 192.0.0.0 to 223.255.255.255. It has a Subnet Mask of 255.255.255.0

Routers primarily are used as traffic directors, forwarding data packets based on their destination IP Address. NAT helps in connecting private network to public network. It is typically used at the border of a personal network. It allows multiple devices on private network to share a single public IP address. Meanwhile NAT provides a additional layer of security by hiding the internal network's private IP address from the public internet. This makes it more difficult for attackers to directly attack the target device. Software that protects against malware attacks, including viruses, worms, ransomware and spyware. This scan devices for malicious code. Encryption of data helps in providing further security.

DOMS | Page No.
Date / /

Firewall acts as a security barrier between your private network and public internet. Hardware and software devices filter the traffic coming in and out based on predefined security rules. Firewall stops unauthorized access attempts and malicious traffic. VPNs helps in establishing secure & private connection.

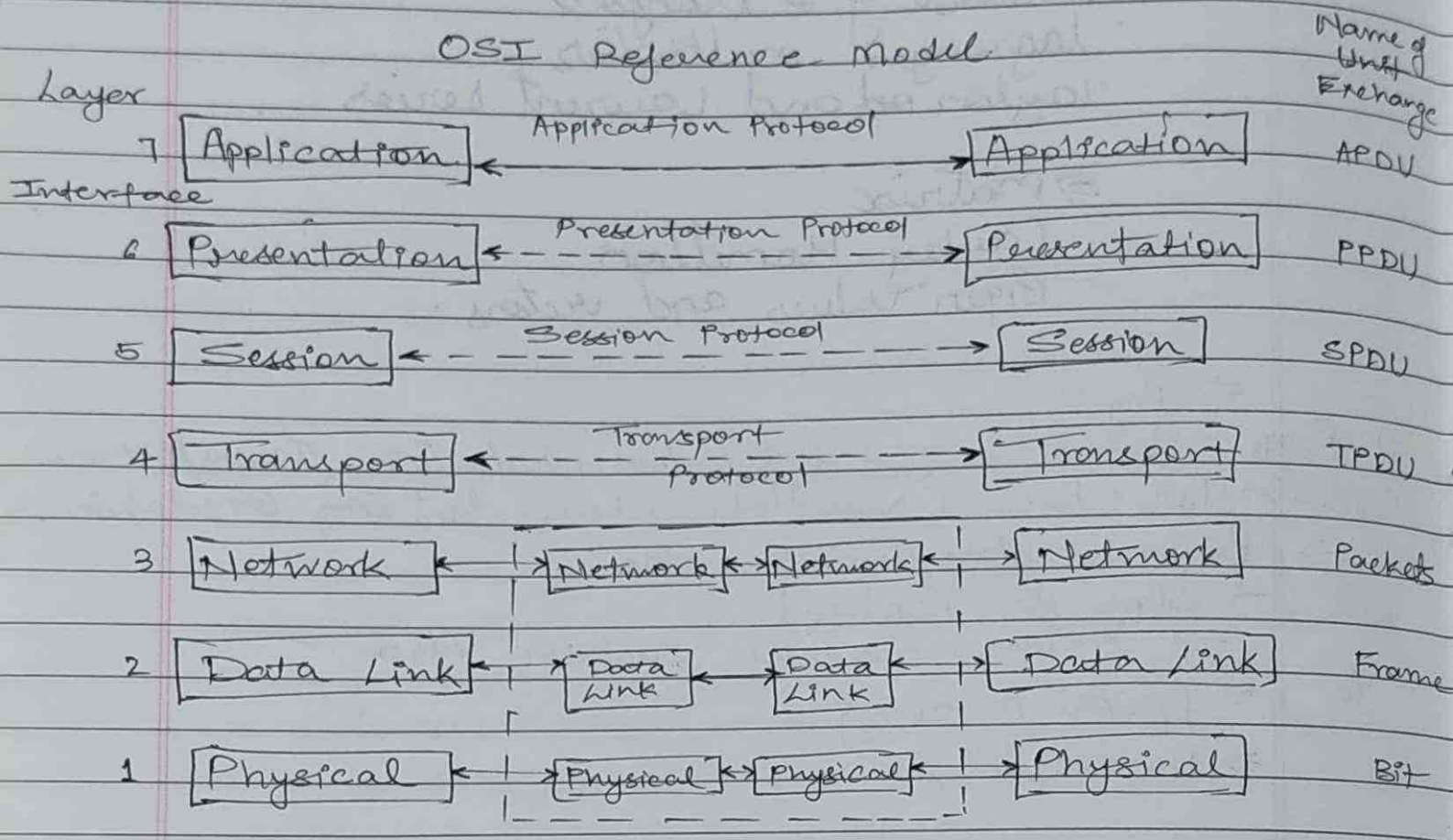
Internet Service Provider needs bandwidth, reliability, uptime for organisation. It offers additional security features like Distributed Denial-of-Service (DDOS). It helps in providing essential connection to wider network. It helps in establishing links & connectival service

Traffic management are critical in ensuring a smooth and efficient network operation. It helps in understanding traffic patterns, bottlenecks and troubleshoot performance issue. It helps in congestion control by shaping bandwidth allocation & queue management. Thus traffic management helps in increasing the productivity by reducing the congestion & cost. Thus increasing the security and and reliability of the network.

KUNAL NEMADE
ROLL NO 89 (S21)
CMND WRITTEN ASSIGNMENT - I

Q.1 Explain ISO/OSI Reference model in detail.

Ans The Internal Standard Organisation (ISO) is a multinational body dedicated to worldwide agreement on international standards. It divides the communication process into seven layer




i) Physical Layer
Converts bits into electronic signal for outgoing messages. Converts electronic signal into bits for incoming messages.

ii) Data Link Layer.
The main task of data link layer is to detect transmission errors. It accomplishes it task by

having the sender break up the input data into data frames and transmits the frame sequentially. At the receiving end, this layer packages raw data from the physical layer into data frames for delivery to the Network Layer.

iii) Network Layer
The network layer counters the operation of the submit to the network layer is responsible for the delivery of individual packets from the source host to the destination host. A key design issue is determing how packets are routed from the source to destination.

iv) Transport Layer
Manages the data transmission accross a network Manages the flow of data bits between parties by segmenting long data streams into smaller data chunks provides acknowledgements of successful transmission and requests resends for packets which arrives with errors.

v) Session layer.
The session layer allows users on different machines, to establish section between them. Various services offer by session layer are dialog control, token management, synchronization

vi) Presentation Layer.
The presentation layer is concerned with the syntax and semantics of the information transmitted.

vii) Application Layer

Application Layer is responsible for providing services to the user. The application layer contains a variety of protocol that are commonly used by user.

Q.2 Write a short note on

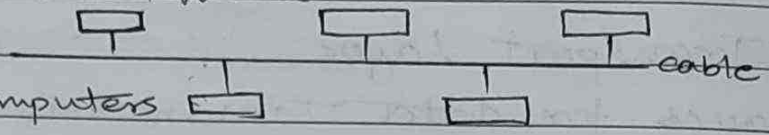
1. Topology 2. IP Addresses

Ans 1. Topology.

Topology defines the physical or logical arrangement of links in a network.

a) Bus Topology :

One long cable acts as a backbone to link all the devices in a network.




Advantages :

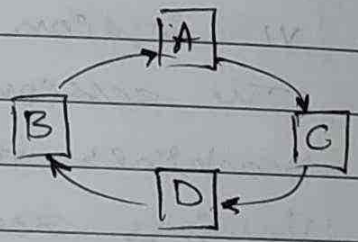
Less expensive, suitable for temporary network

Disadvantages :

Limited Cable length, no fault tolerance.

b) Ring Topology

It is a bus topology in a closed loop. It is a peer-to-peer LAN topology.




Advantages :

Easy to install and configure

All nodes have equal access

Disadvantage :

Inclusion in load leads to decrease in performance no security.

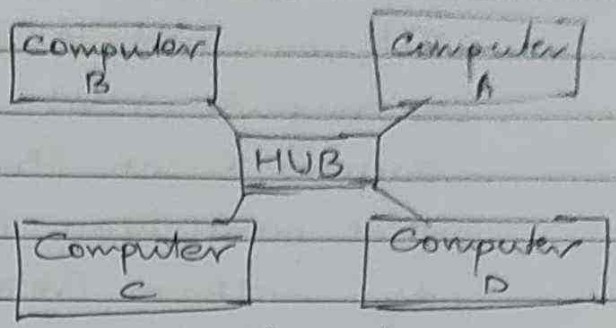
c) Star Topology

Each device has a dedicated point to point view line between only a 'controller' or 'Hub'. The devices are

not directly linked to some other device.

Advantages :

Easy to design and implement. Scalable.




Disadvantages

Each device must be connected to controller. Bottle neck. due to overloaded switch and hub.

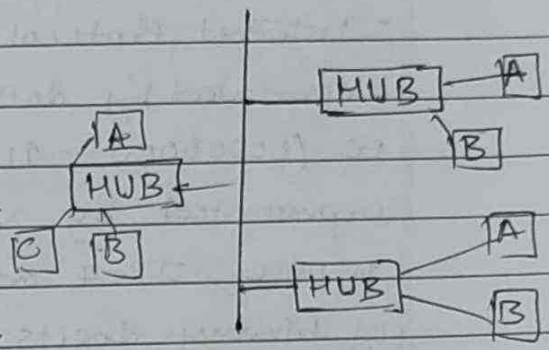
d) Tree Topology.

Tree Topology has some variable from star topology. The nodes in the tree are linked to the central controller.

Advantages :

It allows more devices to be attached in a single controller

It allows the network to priortize the communication




Disadvantages :

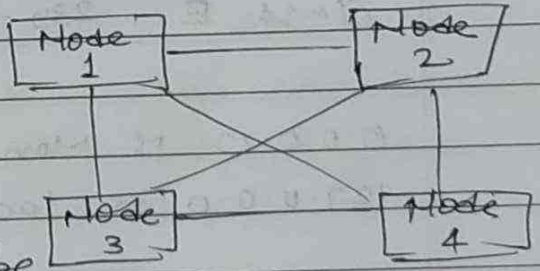
Each device want to be linked to controller. It is expensive, More Installation cost

e) Mesh Topology.

Every device has a direct point to point link between every other device.

Advantages :

It eliminates the traffic problem.




Disadvantage

More number of cables have to be used. Every device must be connected to some other device.

2. IP Addresses.

If you want to download a file from internet, a computer must have an address so that other computers can find and locate mine in order to deliver that particular file or webpage. Address is called as Internet Protocol (IP) Address. Generally device request Internet Service Provider which grants your device across web. It is given from the given range available. Internet activity goes through service provider and route it back to you. IP address are of two types.

a) IPv4

Internet Protocol Version 4 consist of 4 numbers seperated by dots. Each numbe can be from 0-255 ie (00000000 - 11111111). Each number can be represented by a group of 8-digit binary digits. Whole IPv4 address can be represent by 32 bits of binary digits. ie total of 4 million devices. Classes of IPv4 Address are formed for easier management and assignment

| | | |
|---|---|---|
| Class A | 1 - 126 | 126 ($2^7$-2) |
| Class B | 128 - 191 | 16384 |
| Class C | 192-223 | 2097152 |
| Class D | 224-239 | Reserve for Multitasking |
| Class E | 239 - 255 | Reserve for Research & development |

0.0.0.0 is Non-routable address

127.0.0.0 is loopback address

b) IPv6

Problem with IPv4 is we can connect only 4 billion devices uniquely & apparently there are more devices in the world. IPv6 is 128 bit address written in hexadecimal form separated by colons.

Eg: 2011 : 0bd9 : 75c5 : 0000 : 0000 : 6b3e : 0170

Q.3 Differentiate between LAN, MAN, WAN

Ans

| LAN | MAN | WAN |
|---|---|---|
| a) LAN stands for Local Area Network | a) MAN stands for Metropolitan Area Network | a) WAN stands for Wide Area Network. |
| b) Operates in small area such as campus / building | b) Operates in large areas such as cities | b) Operates in larger areas such as country. |
| c) Transmission speed is high. | c) Transmission speed is average | c) Transmission speed is low. |
| d) Propagation delay is short | d) Propagation delay is moderate | d) Propagation delay is long |
| e) Less conjestion | e) More conjestion | e) most conjection |
| f) Design and meintanee is easy | f) Design & meintaine is difficult | f) Design & maintainence is also difficult |
| g) More fault tolerant | g) less fault tolerant | g) Least fault tolerant |
| h) Ownership is private | h) Ownership is private or public | h) Not owned by anyone. |

KUNAL NEMADE
ROLL NO 89 S21
CNND WRITTEN ASSIGNMENT - II

Q.1 Explain the VLAN in detail

Ans Virtual Local Area Network (VLAN) is a way to segment a physical LAN into multiple logical subnetworks. It allows you to create isolated broadcast domains on a single switch, even though the devices are physically connected to same piece of equipment.

Working :

Broadcast Domains - Traditionally a switch forwards all broadcast traffic packets sent to all devices on the network to all of its ports. This can be inefficient and create security issues. VLAN segment the network into smaller broadcast domains, so broadcast traffic only reaches devices within the same VLAN.

Logical Separation - VLAN's logically group devices together regardless of their physical location. For example you can create a VLAN for marketing department, another for finance and another for guest access.

Advantages :

Improved Security : By restricting broadcast traffic VLANs help to isolate sensitive data and prevent unauthorized access.

Increased Performance : By reducing broadcast traffic VLAN can improve network performance.

Simplified Network Management : VLAN's make it easier to manage & troubleshoot network problems.

Greater Flexibility : VLAN's allow you to create a more flexible network that can be easily adapted to changing business model.

6.2 Write a short note on

a) HTTP

Ans Hypertext Transfer Protocol is foundation of communication on World Wide Web. Its a set of rules for how devices exchange information over the internet

Features of HTTP :

Client Server Model : HTTP works in a client server model. A client like a web browser initiates a request to a server like a webpage. The server process the request and sends back the response.

Stateless : HTTP is stateless, meaning each request-response interaction is independent. The server doesn't remember previous interactions with client

Request & Response : An HTTP request includes the type of request as retrieve a webpage, the address of resource and additional information. Response contains a status code eg 200 for success, headers with details about the content and itself

Applications : It is primarily used in web browsing. It is also used where we exchange data over the internet.

b) FTP

Ans File Transfer Protocol (FTP) is a workhouse of the internet, enabling the transfers of files between computers on a network. Some features are

Function : FTP allows you to upload and download files from remote server

Client-Server model : FTP client connects to FTP server on another machine.

Separate Connections : FTP utilizes two connection a control channel for sending commands like login

and file requests and data channel for transferring actual files data
Security : Traditionally, FTP transmits data unencrypted, making it less secure for sensitive information. Secure alternatives like SFTP
Application :
Web developers use it to upload and manage website files on web servers.
It facilitates transferring large files that might be hard to transfer through email
System administrators use it for server maintainence & file backups

c) DNS

Dns Domain Name System is called the Address Book of the internet. It acts like internet's phonebook. It translates human-readable website address like google.com to numerical Ip Address that computers used to connect each other. DNS is a fundamental service that runs invisibly in the background, ensuring smooth and user-friendly web browsing experience.
Functions : DNS resolves domain names in Ip Address. It makes internet user friendly by using in names instead of complex numbers
Backscene : When you enter a domain name in the browser, a DNS server is contacted in the background. Server looks up the corresponding Ip Address and directs your browser to right website
Distributed System : DNS is hierarchical and distributed system, with multiple servers working together to handle requests efficiently

Analogy : DNS is like a phonebook where website names are listed alphabetically and their corresponding IP address are like phone numbers

d) SNMP

Simple Network Management Protocol (SNMP) is a protocol for network monitoring and managements. It is widely used and valuable tool for network administrators to maintain the health and performance of their network

Function : It facilitates the exchange of information between network devices and a central management system.

Benefits : It enable network administrators to monitor device health and performance like CPU usage and memory. Identify and troubleshoot network issues. Manage device configuration.

Components : It operates on a client - server model. Agent has software running on network devices that collects and stores management info Manager is a network management system that sends requests to agents, retrieves information and present it for monitoring and analysis. Management Information Base is standardized database defining the type of information agents can store and retrieve

Communication : It uses UDP ports 161 and 162 for communication between agents and managers

Security : snmp offers basic community string authentication, it essential to implement stronger security measures for sensitive network environments

e) SMTP

Ans Simple Main Transfer Protocol (SMTP) is an unseen force behind sending email. It dictates how email clients and servers communicate to deliver your messages across the Internet

Function : SMTP acts like a post office for emails ensuring proper delivery from sender to receiver

Client-Server Model : Following a client server model Email clients like outlook or gmail acts a clientes while mail servers handle the server side communication

Transmission Process : On sending an email, client establishes a connection with mail server using port 25. Email data, including sender, receipient and message content is then transmitted using series of text based commands and responses

Push Protocol : Unlike protocols for retrieving emails, SMTP is a push protocol. It focuses solely on sending email from client to server

Security : SMTP is relatively simple, email security is crucial. Authentication and encryption mechanisms and often implemented to protect emails from unauthorized access.