Name : Pritam Rao
Branch : TE Computer
Batch : D
UID : 2018130044

Lab 2: Basic Network Utilities

This lab introduces some basic network monitoring/analysis tools. There are a few exercises along the way. You should write up answers to the ***ping*** and ***traceroute*** exercises and turn them in next lab. (You should try out each tool, whether it is needed for an exercise or not!).

Prerequisite: Basic understanding of command line utilities of Linux Operating system.

Some Basic command line Networking utilities

Start with a few of the most basic command line tools. These commands are available on Unix, including Linux (and the first two, at least, are also for Windows). Some parameters or options might differ on different operating systems. Remember that you can use man <command> to get information about a command and its options.

ping — The command ping <host> sends a series of packets and expects to receieve a response to each packet. When a return packet is received, ping reports the round trip time (the time between sending the packet and receiving the response). Some routers and firewalls block ping requests, so you might get no reponse at all. Ping can be used to check whether a computer is up and running, to measure network delay time, and to check for dropped packets indicating network congestion. Note that <host> can be either a domain name or an IP address. By default, ping will send a packet every second indefinitely; stop it with Control-C

Experiments with Ping

1. Ping the any hosts 10 times (i.e., packet count is 10) with a packet size of 64 bytes, 100 bytes, 500 bytes, 1000 bytes, 1400 bytes

```
> ping -c 10 -s 64 google.com
PING google.com (142.250.67.174) 64(92) bytes of data.
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=1 ttl=118 time=5.01 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=2 ttl=118 time=7.30 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=3 ttl=118 time=6.92 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=4 ttl=118 time=3.44 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=5 ttl=118 time=13.4 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=6 ttl=118 time=6.73 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=7 ttl=118 time=10.4 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=8 ttl=118 time=6.65 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=9 ttl=118 time=6.56 ms
72 bytes from bom12s07-in-f14.1e100.net (142.250.67.174): icmp_seq=10 ttl=118 time=7.17 ms
```

```
--- google.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9015ms
rtt min/avg/max/mdev = 3.440/7.360/13.419/2.619 ms
> ping -c 10 -s 100 google.com
PING google.com (142.250.67.174) 100(128) bytes of data.

--- google.com ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9014ms

> ping -c 10 -s 500 google.com
PING google.com (142.250.67.174) 500(528) bytes of data.

--- google.com ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9014ms

>
```

```
> ping -c 10 -s 1000 www.uw.edu
PING www.washington.edu (128.95.155.198) 1000(1028) bytes of data.
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=1 ttl=46 time=285 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=2 ttl=46 time=308 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=3 ttl=46 time=342 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=4 ttl=46 time=352 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=5 ttl=46 time=273 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=6 ttl=46 time=295 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=7 ttl=46 time=317 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=8 ttl=46 time=352 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=9 ttl=46 time=260 ms
1008 bytes from www4.cac.washington.edu (128.95.155.198): icmp_seq=10 ttl=46 time=282 ms

--- www.washington.edu ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9008ms
rtt min/avg/max/mdev = 260.259/306.784/352.079/31.591 ms
> ping -c 10 -s 1400 www.mozilla.org
PING www.mozilla.org.cdn.cloudflare.net (104.18.164.34) 1400(1428) bytes of data.
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=1 ttl=58 time=4.74 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=2 ttl=58 time=5.45 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=3 ttl=58 time=7.67 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=4 ttl=58 time=20.0 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=5 ttl=58 time=7.38 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=6 ttl=58 time=7.88 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=7 ttl=58 time=7.43 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=8 ttl=58 time=7.42 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=9 ttl=58 time=15.5 ms
1408 bytes from 104.18.164.34 (104.18.164.34): icmp_seq=10 ttl=58 time=7.80 ms

--- www.mozilla.org.cdn.cloudflare.net ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9014ms
rtt min/avg/max/mdev = 4.742/9.128/20.021/4.542 ms
```

Q. Does the average RTT vary between different hosts? What aspects of latency (transmit, propagation, and queueing delay) might impact this and why?

```
> ping -c 10 -s 64 facebook.com
PING facebook.com (157.240.16.35) 64(92) bytes of data.
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=1 ttl=56 time=3.58 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=2 ttl=56 time=6.21 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=3 ttl=56 time=6.50 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=4 ttl=56 time=6.50 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=5 ttl=56 time=6.95 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=6 ttl=56 time=6.39 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=7 ttl=56 time=6.30 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=8 ttl=56 time=6.31 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=9 ttl=56 time=6.34 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=10 ttl=56 time=6.20 ms
```

```
> ping -c 10 -s 64 www.uw.edu
PING www.washington.edu (128.95.155.135) 64(92) bytes of data.
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=1 ttl=46 time=289 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=2 ttl=46 time=256 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=3 ttl=46 time=336 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=4 ttl=46 time=259 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=5 ttl=46 time=278 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=6 ttl=46 time=301 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=7 ttl=46 time=265 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=8 ttl=46 time=346 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=9 ttl=46 time=267 ms
72 bytes from www2.cac.washington.edu (128.95.155.135): icmp_seq=10 ttl=46 time=289 ms

--- www.washington.edu ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9011ms
rtt min/avg/max/mdev = 255.955/288.671/346.331/29.620 ms
```

From the above figures, we can clearly conclude that the RTT is dependent on the host on which the ‘ping’ command is used.

Propagation delay is the time taken by the first bit to travel from sender to receiver end.Factors on which propagation delay depends are **distance** and **propagation speed**. So, there exists a propagation delay in the two cases.

Queueing delay is the time difference between when the packet arrived at its destination and when the packet data was processed or executed. It depends on the **number of packets, size of the packet** and **bandwidth** of the network.

Q.Does the average RTT vary with different packet sizes? What aspects of latency (transmit, propagation, and queueing delay) might impact this and why?

```
> ping -c 10 -s 512 facebook.com
PING facebook.com (157.240.16.35) 512(540) bytes of data.
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=1 ttl=56 time=3.63 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=2 ttl=56 time=3.76 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=3 ttl=56 time=6.67 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=4 ttl=56 time=6.65 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=5 ttl=56 time=3.75 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=6 ttl=56 time=8.82 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=7 ttl=56 time=4.94 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=8 ttl=56 time=7.80 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=9 ttl=56 time=4.81 ms
520 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=10 ttl=56 time=6.70 ms
```

```
> ping -c 10 -s 64 facebook.com
PING facebook.com (157.240.16.35) 64(92) bytes of data.
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=1 ttl=56 time=8.31 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=2 ttl=56 time=10.6 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=3 ttl=56 time=5.86 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=4 ttl=56 time=5.79 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=5 ttl=56 time=4.25 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=6 ttl=56 time=6.84 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=7 ttl=56 time=16.3 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=8 ttl=56 time=3.71 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=9 ttl=56 time=8.80 ms
72 bytes from edge-star-mini-shv-01-bom1.facebook.com (157.240.16.35): icmp_seq=10 ttl=56 time=10.1 ms

--- facebook.com ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9016ms
rtt min/avg/max/mdev = 3.714/8.047/16.294/3.515 ms
```

From the above pictures , its clear that average RTT varies with different packet sizes. This is because of the propogation delay and the queiueng delay.

Exercise 1: Experiment with ping to find the round trip times to a variety of destinations. Write up any interesting observations, including in particular how the round trip time compares to the physical distance. Here are few places from who to get replies: www.uw.edu, www.cornell.edu, berkeley.edu, www.uchicago.edu, www.ox.ac.uk (England), www.u-tokyo.ac.jp (Japan).

From the images above following cnclusions are made :

- The length a signal has to travel correlates with the time taken for a request to reach a server.

➢ Intermediate routers or servers take time to process a signal, increasing RTT. The more hops a signal has to travel through, the higher the RTT.

➢ RTT typically increases when a network is congested with high levels of traffic. Conversely, low traffic times can result in decreased RTT.

nslookup — The command nslookup <host> will do a DNS query to find and report the IP address (or addresses) for a domain name or the domain name corresponding to an IP address. To do this, it contacts a "DNS server." Default DNS servers are part of a computer's network configuration. (For a static IP address in Linux, they are configured in the file /etc/network/interfaces that you encountered in the last lab.) You can specify a different DNS server to be used by nslokup by adding the server name or IP address to the command: nslookup <host> <server>

```
> nslookup www.spit.ac.in
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   www.spit.ac.in
Address: 43.252.193.19

> nslookup google.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 172.217.166.174
Name:   google.com
Address: 2404:6800:4009:812::200e
```

ifconfig — You used ifconfig in the previous lab. When used with no parameters, ifconfig reports some information about the computer's network interfaces. This usually includes lo which stands for localhost; it can be used for communication between programs running on the same computer. Linux often has an interface named eth0, which is the first ethernet card. The information is different on Mac OS and Linux, but includes the IP or "inet" address and ethernet or "hardware" address for an ethernet card. On Linux, you get the number of packets received (RX) and sent (TX), as well as the number of bytes transmitted and received. (A better place to monitor network bytes on our Linux computers is in the GUI program System Monitor, if it is installed!!!.)

```
> ifconfig
enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 54:48:10:b3:3f:6a  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

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 6455  bytes 627003 (627.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6455  bytes 627003 (627.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.0.105  netmask 255.255.255.0  broadcast 192.168.0.255
        inet6 fe80::f1de:177b:2e9d:6984  prefixlen 64  scopeid 0x20<link>
        ether 90:32:4b:2d:1f:bf  txqueuelen 1000  (Ethernet)
        RX packets 152640  bytes 117267768 (117.2 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 106501  bytes 23777561 (23.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

>
```

netstat — The netstat command gives information about network connections. I often use netstat -t -n which lists currently open TCP connections (that's the "-t" option) by IP address rather than domain name (that's the "-n" option). Add the option "-l" (lower case ell) to list listening sockets, that is sockets that have been opened by server programs to wait for connection requests from clients: netstat -t -n -l. (On Mac, use netstat -p tcp to list tcp connections, and add "-a" to include listening sockets in the list.)

```
> netstat -t -n -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN
tcp6       0      0 ::1:631                 :::*                    LISTEN
> netstat -t -n
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.0.105:33394     74.125.68.188:5228      ESTABLISHED
tcp        0      0 192.168.0.105:36500     157.240.16.52:443       ESTABLISHED
tcp        0      0 192.168.0.105:41788     52.32.142.97:443        ESTABLISHED
tcp        0      0 192.168.0.105:42486     13.227.141.14:443       ESTABLISHED
tcp        0      0 192.168.0.105:59814     34.213.232.243:443      ESTABLISHED
tcp        0      0 192.168.0.105:60606     104.17.79.107:443       ESTABLISHED
tcp        0      0 192.168.0.105:60120     52.108.236.4:443        ESTABLISHED
tcp        0      0 192.168.0.105:35042     151.101.193.44:80       ESTABLISHED
tcp        1      1 192.168.0.105:37904     34.213.232.243:80       LAST_ACK
tcp        0      0 192.168.0.105:46112     172.217.166.174:443     ESTABLISHED
tcp        0      0 192.168.0.105:58596     52.55.211.134:443       ESTABLISHED
```

telnet — Telnet is an old program for remote login. It's not used so much for that any more, since it has no security features. But basically, all it does is open a connection to a server and allow server and client to send lines of plain text to each other. It can be used to check that it's possible to connect to a server and, if the server communicates in plain text, even to interact with the server by hand. Since the Web uses a plain text protocol, you can use telnet to connect to a web client and play the part of the web browser. I will suggest that you to do this with your own web server when you write it, but you might want to try it now. When you use telnet in this way, you need to specify both the host and the port number to which you want to connect: telent <host> <port>. For example, to connect to the web server on www.spit.ac.in: telnet spit.ac.in 80

Output:
> telnet www.spit.ac.in 8000
Trying 43.252.193.19...
telnet: Unable to connect to remote host: Connection timed out

traceroute — Traceroute is discussed in man utility. The command traceroute <host> will show routers encountered by packets on their way from your computer to a specified <host>. For each n = 1, 2, 3,..., traceroute sends a packet with "time-to-live" (ttl) equal to n. Every time a router forwards a packet, it decreases the ttl of the packet by one. If the ttl drops to zero, the router discards the packet and sends an error message back to the sender of the packet. (Again, as with ping, the packets might be blocked or might not even be sent, so that the error messages will never be received.) The sender gets the identity of the router from the source of the error message. Traceroute will send packets until n reaches some set upper bound or until a packet actually gets through to the destination. It actually does this three times for each n. In this way, it identifies routers that are one step, two steps, three steps, ... away from the source computer. A packet for which no response is received is indicated in the output as a *.

```
> traceroute cs.manchester.ac.uk
traceroute to cs.manchester.ac.uk (130.88.101.49), 64 hops max
  1   192.168.0.1  2.585ms  1.530ms  2.064ms
  2   5.5.5.3  3.098ms  66.116ms  22.555ms
  3   10.200.100.254  43.940ms  15.399ms  28.062ms
  4   45.126.169.209  29.800ms  45.013ms  131.810ms
  5   *  *  *
  6   182.73.199.157  6.422ms  6.642ms  4.154ms
  7   182.79.154.0  328.683ms  204.624ms  204.568ms
  8   *  *  *
  9   62.115.175.131  204.043ms  204.028ms  211.668ms
 10   146.97.35.197  207.607ms  137.058ms  *
 11   146.97.33.2  179.923ms  198.773ms  232.181ms
 12   146.97.33.22  239.333ms  144.003ms  160.918ms
 13   146.97.33.42  202.668ms  207.298ms  210.330ms
 14   146.97.38.42  199.385ms  208.665ms  603.103ms
 15   *  *  *
 16   130.88.249.194  142.505ms  158.627ms  *
 17   *  *  *
 18   *  *  *
 19   130.88.101.49  171.996ms  210.843ms  201.852ms
```

Exercise 2: (Very short.) Use traceroute to trace the route from your computer to math.hws.edu and to www.hws.edu. Explain the difference in the results.

```
> traceroute math.hws.edu
traceroute to math.hws.edu (64.89.144.237), 64 hops max
  1   192.168.0.1  2.613ms  1.834ms  1.962ms
  2   5.5.5.3  2.592ms  2.374ms  2.954ms
  3   10.200.100.254  2.628ms  2.869ms  2.269ms
  4   45.126.169.209  2.926ms  2.851ms  2.551ms
  5   103.59.200.254  16.142ms  3.391ms  3.028ms
  6   182.73.199.157  6.718ms  6.554ms  6.827ms
  7   182.79.234.217  281.759ms  329.523ms  284.216ms
  8   4.26.0.17  330.491ms  265.379ms  325.283ms
  9   *  *  4.69.207.49  261.885ms
 10   *  *  *
 11   35.248.1.158  379.082ms  307.125ms  307.073ms
 12   66.195.65.170  306.986ms  307.060ms  409.736ms
 13   64.89.144.100  295.130ms  318.796ms  307.032ms
 14   *  *  *
 15   *  *  *
 16   *  *  *
 17   *  *  *
 18   *  *  *
 19   *  *  *
```

```
> traceroute www.hws.edu
traceroute to www.hws.edu (64.89.145.159), 64 hops max
  1   192.168.0.1  2.090ms  2.022ms  2.031ms
  2   5.5.5.3  4.896ms  3.698ms  2.934ms
  3   10.200.100.254  5.127ms  3.400ms  3.480ms
  4   45.126.169.209  3.038ms  2.883ms  2.448ms
  5   103.59.200.254  3.997ms  4.209ms  5.620ms
  6   182.73.199.157  17.172ms  7.477ms  7.354ms
  7   182.79.245.81  327.236ms  302.848ms  306.497ms
  8   4.26.0.89  305.995ms  307.597ms  255.710ms
  9   *  *  *
 10   *  *  *
 11   35.248.1.158  361.349ms  307.158ms  307.294ms
 12   66.195.65.170  279.902ms  333.913ms  307.089ms
 13   64.89.144.100  307.104ms  409.402ms  307.275ms
 14   *  *  *
 15   *  *  *
 16   *  *  *
 17   *  *  *
 18   *  *  *
 19   *  *  *
 20   *  *  *
```

From the above images, the first row shows that the process of route tracing The next six rows in both the cases are similar as the route is being traced starting from the ISP (Internet service provider) of the user. The next rows after 6th router clearly show that the route is completely different

Exercise 3: Two packets sent from the same source to the same destination do not necessarily follow the same path through the net. Experiment with some sources that are fairly far away. Can you find cases where packets sent to the same destination follow different paths? How likely does it seem to be? What about when the packets are sent at very different times? Save some of the outputs from traceroute. (You can copy them from the Terminal window by highlighting and right-clicking, then paste into a text editor.) Come back sometime next week, try the same destinations again, and compare the results with the results from today. Report your observations.

```
> traceroute www.umich.edu
traceroute to www.umich.edu (141.211.243.251), 64 hops max
  1   192.168.0.1  2.493ms  1.899ms  4.574ms
  2   5.5.5.3  2.906ms  2.300ms  2.410ms
  3   10.200.100.254  3.268ms  2.804ms  2.406ms
  4   45.126.169.209  2.622ms  2.994ms  2.738ms
  5   103.59.200.254  3.321ms  2.837ms  3.220ms
  6   182.73.199.157  4.028ms  3.730ms  3.964ms
  7   182.79.224.181  55.863ms  58.020ms  59.575ms
  8   63.218.107.193  189.829ms  204.583ms  204.733ms
  9   63.223.43.102  307.018ms  307.017ms  242.687ms
 10   63.223.43.110  268.927ms  240.438ms  271.248ms
 11   *  *  *
 12   *  *  *
 13   *  *  *
 14   *  *  *
 15   *  *  *
 16   64.57.20.244  399.601ms  409.423ms  307.084ms
 17   64.57.20.244  409.464ms  409.482ms  409.358ms
 18   64.57.29.178  409.482ms  409.329ms  409.709ms
 19   192.12.80.70  409.464ms  409.352ms  409.510ms
 20   192.12.80.25  409.553ms  409.181ms  409.492ms
 21   192.12.80.25  409.268ms  419.763ms  399.420ms
 22   141.211.0.142  409.465ms  409.414ms  409.417ms
 23   141.211.0.150  426.785ms  404.384ms  532.828ms
 24   198.108.13.61  616.615ms  353.197ms  342.038ms
 25   141.211.243.251  284.593ms  321.668ms  434.616ms
```

QUESTIONS ABOUT PATHS

Now look at the results you gathered and answer the following questions about the paths taken by your packets. Store your answers in a file named raceroute.txt.

1. Is any part of the path common for all hosts you tracerouted?

Yes, from the starting path till address 182.73.199.157 i.e till 6^{th} router is common at both times.

2. Is there a relationship between the number of nodes that show up in the traceroute and the location of the host? If so, what is this relationship?

The number of nodes involved depend on the bandwith and the traffic of the network and also if the distance between the user and the destination host is more then more number of nodes will be involved in the traceroute.

3. Is there a relationship between the number of nodes that show up in the traceroute and latency of the host (from your ping results above)? Does the same relationship hold for all hosts?

Yes, if the latency is involved then traceroute request gets timed out after certain maximum hops but the same relationship will not hold for all hosts.

Whois — The *whois* command can give detailed information about domain names and IP addresses. If it is not installed on the computers then install it with command sudo apt-get install whois in. *Whois* can tell you what organization owns or is responsible for the name or address and where to contact them. It often includes a list of domain name servers for the organization.

When using *whois* to look up a domain name, use the simple two-part network name, not an individual computer name (for example, *whois spit.ac.in*).

<u>Exercise 4:</u> (Short.) Use *whois* to investigate a well-known web site such as google.com or amazon.com, and write a couple of sentences about what you find out.

```
> whois google.com
   Domain Name: GOOGLE.COM
   Registry Domain ID: 2138514_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.markmonitor.com
   Registrar URL: http://www.markmonitor.com
   Updated Date: 2019-09-09T15:39:04Z
   Creation Date: 1997-09-15T04:00:00Z
   Registry Expiry Date: 2028-09-14T04:00:00Z
   Registrar: MarkMonitor Inc.
   Registrar IANA ID: 292
   Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
   Registrar Abuse Contact Phone: +1.2083895740
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2020-08-31T13:23:07Z <<<

For more information on Whois status codes, please visit https://icann.org/epp

NOTICE: The expiration date displayed in this record is the date the
registrar's sponsorship of the domain name registration in the registry is
currently set to expire. This date does not necessarily reflect the expiration
date of the domain name registrant's agreement with the sponsoring
registrar.  Users may consult the sponsoring registrar's Whois database to
view the registrar's reported date of expiration for this registration.
```

From the image above we get Domain Name, Registry domain id , registrar url, Domain status and server name.

CONCLUSION:
I learnt about basic network utilities and what they are used for and executed the commands for same.