

# CS 252: Lab 2

Abhinav Gupta, Gurnoor Singh Khurana, Sambit Behera  
190050003, 190050045, 190050104

May 11, 2021

## Network Diagnostic Tools

### Key

Measurement 1 (**M1**): On a Mac device by Gurnoor Singh Khurana, located in Amritsar, Punjab.  
Measurement 2 (**M2**): On a Windows device by Abhinav Gupta, located in Ghaziabad, UP.  
Measurement 3 (**M3**): On a Mac device by Sambit Behera, located in Bhubaneswar, Orissa.

### Solutions/Measurements

#### (i) `ifconfig`

Number of bits used in IPv4, IPv6 and MAC(hardware) address are 32, 128 and 48 respectively. Some retailers are also rolling out 64-bit MAC addresses. MTU stands for Maximum Transmission Unit. It is measured in bytes (1 byte = 8 bits). It is the largest *protocol data unit* that can be communicated in a single network layer transaction.

Transmit Queue Length is measured in the number of packets per kernel. It is the maximum number of packets allowed in the transmit queue of a Network Interface Device(NIC).

#### M1

- IPv4 address: 192.168.29.183
- IPv6 address: fe80::49e:1ec9:1f97:1ffe
- Hardware Address (MAC): a4:83:e7:34:ba:df
- Transmit Queue Length : **Receiving**: 128 packets **Sending**: 256 packets
- MTU: 1500

#### M2

- IPv4 address- **Ethernet**: 192.168.1.9, **WiFi**: 192.168.1.4
- IPv6 address- **Ethernet**: fe80:0:0:b361:bb1c:5323:a8d6 (or fe80::b361:bb1c:5323:a8d6 in condensed format), **WiFi**: fe80:0:0:3533:11f7:130b:fcf (or fe80::3533:11f7:130b:fcf in condensed format)
- Hardware address- **Ethernet**: 80:e8:2c:c6:b5:cc, **WiFi**: d0:ab:d5:c8:ac:30
- Transmit Queue Length- 1000 packets for both interfaces
- MTU: 1464 (found pinging packets of different sizes until they fragment, basically binary search)

#### M3

For Wifi :-

- IPv4 address: 192.168.1.3
- IPv6 address: fe80::420:b225:94c2:80c4
- Hardware address (MAC)- 38:f9:d3:ec:56:bd
- Transmit Queue length- **Receiving**: 128 packets **Sending**: 256 packets
- MTU: 1500

(ii) **traceroute**

Here is number of hops along with average RTTs from our devices M1, M2, M3 and from 5 different geographical locations G1, ..., G5 ({hops, RTT(in ms)} format).

Locations are, namely (a) **G1**: Madrid, (b) **G2**: Brisbane, (c) **G3**: Johannesburg, (d) **G4**: Montreal; and (e) **G5**: Hong Kong

|    | Google   | CNN      | IITD       |
|----|----------|----------|------------|
| M1 | 11, 14.3 | 14, 37.7 | 23, 21.3   |
| M2 | 8, 4.3   | 6, 3.7   | 13, 6.7    |
| M3 | 11, 30.6 | —        | —          |
| G1 | 7, 0     | 4, 0     | 14, 135    |
| G2 | 12, 11.5 | 6, 0     | 26, 174    |
| G3 | 8, 15    | 4, 0     | 24, 227.75 |
| G4 | 9, 1     | 13, 8    | 25, 227.25 |
| G5 | 22, 12   | 10, 2.25 | 17, 223    |

Figure 1

Here are the destination IP addresses of the servers (rather websites) when accessed from these locations.

|    | Google          | CNN            | IITD        |
|----|-----------------|----------------|-------------|
| M1 | 172.217.24.228  | 151.101.153.67 | 103.27.9.24 |
| M2 | 216.58.196.196  | 199.232.21.67  | 103.27.9.24 |
| M3 | 142.250.71.4    | 151.101.153.67 | 103.27.9.24 |
| G1 | 216.58.209.68   | 151.101.133.67 | 103.27.9.24 |
| G2 | 172.217.167.100 | 151.101.97.67  | 103.27.9.24 |
| G3 | 172.217.170.68  | 151.101.225.67 | 103.27.9.24 |
| G4 | 172.217.13.164  | 151.101.125.67 | 103.27.9.24 |
| G5 | 108.177.97.99   | 151.101.77.67  | 103.27.9.24 |

Figure 2

**Note:** '—' is given when we reach a firewall somewhere in the path to server but destination is anyways shown so IP table is filled.

We see 0 readings in the RTT table. We are not sure how (because earlier hops had non-zero timings), but it takes server very less time to reach the web-address specified.

We can observe that the IP addresses are different for the same domain names(or websites). This is because websites tend to have multiple IP addresses hosted at multiple locations, mostly for load balancing and redundancy. The DNS server decides which IP address to return when we ping the domain name. When provided with multiple options, the DNS server can return any of the available IP addresses. The command `dig <domain name>` on Linux/Mac will return information about the DNS server and available IP addresses for that location.

Also, for the IITD website, destination IP address is same no matter what the source. This is most likely because they have only one server hosting their website at a single location.

(iii) **ping**

| Continent     | Website   | RTT(in <i>ms</i> ) |     |     |                                       |
|---------------|---|--------------------|-----|-----|---------------------------------------|
|               |   | M1                 | M2  | M3  | <a href="https://ping.eu">ping.eu</a> |
| Australia     | <a href="https://sydney.edu.au">sydney.edu.au</a>           | 179                | 429 | 357 | 293                                   |
| Asia          | <a href="https://iitd.ac.in">iitd.ac.in</a>                 | 23                 | 7   | 120 | 155                                   |
| North America | <a href="https://mit.edu">mit.edu</a>                       | 42                 | 41  | 284 | 159                                   |
| South America | <a href="https://fearp.usp.br">fearp.usp.br</a>             | 485                | 372 | 453 | 240                                   |
| Africa        | <a href="https://gov.za">gov.za</a>                         | 364                | 302 | 485 | 180                                   |
| Europe        | <a href="https://tu-braunschweig.de">tu-braunschweig.de</a> | 169                | 150 | 190 | 17                                    |

Figure 3: Average RTTs for different geographical locations

**Observations:**

- (a) RTT increases as the geographical distance between client and server increases. For measurements from our machines, RTT values to [iitd.ac.in](https://iitd.ac.in) are the least while for <https://ping.eu>, RTT values to [tu-braunschweig.de](https://tu-braunschweig.de) are the least. This clearly follows from the fact that we are located in Asia and servers of <https://ping.eu> are located in Europe.
- (b) For different packets, even from the same location, RTT times are different. This is because the network traffic changes continuously and the traffic at a given point determines the RTT.

(iv) **iperf**

| Server  |     | Clients |      |      |
|---|-----|---------|------|------|
|   |     | M1      | M2   | M3   |
| <a href="https://iperf.scottlinux.com">iperf.scottlinux.com</a> | TCP | 2.42    | 2.08 | 31.4 |
|   | UDP | 90.8    | 96.3 | 82.4 |
| <a href="https://ping.online.net">ping.online.net</a>           | TCP | 8.59    | 9.32 | 42.9 |
|   | UDP | 95.8    | 96.3 | 85.8 |

Figure 4: Protocol-wise bandwidths(in Mbps) on different PCs (on two servers)

Exact UDP bandwidths are measured by setting parameter `-b` to `0M` (this was discovered from the man page of `iperf`). Final command being

```
iperf3 -c <server> -u -b 0M
```

As expected, the bandwidths for UDP are 2-10 times the TCP bandwidths in each measurement. This is because there is no packet loss check provided by the UDP protocol, leading to a faster but inaccurate data transmission. All three measurements give the same expected results.

**Bonus:**

To setup the `iperf3` server in the Windows PC, following steps were followed:

- Made IP address static in Network Settings, instead of original dynamic IP. This was done by overriding the automatic IP allocation system of the LAN.
- Forwarded port 5201 from router to my PC by going on router setup page, so that all requests coming to that port on my router(public IP) gets redirected to my PC. There was nothing specific with port 5201 but it was the default port captured by `iperf3` application.
- On the PC, enabled port 5201 for outside communication by setting up firewall rules for TCP and UDP protocols separately.

| Server |     | Clients |      |
|--------|-----|---------|------|
|        |     | M1      | M3   |
| M2     | TCP | 22.4    | 31.2 |
|        | UDP | 92.9    | 86.3 |

Figure 5: Protocol-wise bandwidths (in Mbps) (on M2 as server)

Screenshots of server as well as client side for both TCP and UDP protocols, results are same i.e. higher speeds but also high packet loss in case of UDP as compared to TCP. Throughout the experimentation, M2 was the server.

```
Server listening on 5201
Accepted connection from 49.36.228.100, port 55201
[ 5] local 192.168.1.29 port 5201 connected to 49.36.228.100 port 55202
[ ID] Interval      Transfer      Bandwidth
[ 5] 0.00-1.00 sec  1.56 MBytes  13.1 Mbits/sec
[ 5] 1.00-2.00 sec  2.27 MBytes  19.0 Mbits/sec
[ 5] 2.00-3.00 sec  2.23 MBytes  18.7 Mbits/sec
[ 5] 3.00-4.01 sec  2.36 MBytes  19.5 Mbits/sec
[ 5] 4.01-5.00 sec  2.16 MBytes  18.4 Mbits/sec
[ 5] 5.00-6.01 sec  2.40 MBytes  19.9 Mbits/sec
[ 5] 6.01-7.01 sec  2.14 MBytes  18.1 Mbits/sec
[ 5] 7.01-8.01 sec  2.43 MBytes  20.4 Mbits/sec
[ 5] 8.01-9.00 sec  2.14 MBytes  18.0 Mbits/sec
[ 5] 9.00-10.00 sec  2.37 MBytes  19.9 Mbits/sec
[ 5] 10.00-10.07 sec 179 KBytes  21.6 Mbits/sec
[ ID] Interval      Transfer      Bandwidth
[ 5] 0.00-10.07 sec  0.00 Bytes   0.00 bits/sec
[ 5] 0.00-10.07 sec  22.2 MBytes  18.5 Mbits/sec
sender
receiver
```

(a) Server: M2-M1 connection

```
root@docker-desktop:/# iperf3 -c 171.76.13.170
Connecting to host 171.76.13.170, port 5201
[ 5] local 192.168.65.3 port 59448 connected to 171.76.13.170 port 5201
[ ID] Interval      Transfer      Bitrate      Retr      Cwnd
[ 5] 0.00-1.00 sec  2.41 MBytes  20.2 Mbits/sec  0          78.4 KBytes
[ 5] 1.00-2.00 sec  2.21 MBytes  18.5 Mbits/sec  0          97.0 KBytes
[ 5] 2.00-3.00 sec  2.76 MBytes  23.1 Mbits/sec  0          135 KBytes
[ 5] 3.00-4.00 sec  2.14 MBytes  18.0 Mbits/sec  0          111 KBytes
[ 5] 4.00-5.00 sec  2.45 MBytes  20.6 Mbits/sec  0          138 KBytes
[ 5] 5.00-6.00 sec  2.14 MBytes  18.0 Mbits/sec  0          124 KBytes
[ 5] 6.00-7.00 sec  2.33 MBytes  19.5 Mbits/sec  0          160 KBytes
[ 5] 7.00-8.00 sec  2.21 MBytes  18.5 Mbits/sec  0          168 KBytes
[ 5] 8.00-9.00 sec  2.57 MBytes  21.6 Mbits/sec  0          180 KBytes
[ 5] 9.00-10.00 sec  2.27 MBytes  19.0 Mbits/sec  0          187 KBytes
[ ID] Interval      Transfer      Bitrate      Retr
[ 5] 0.00-10.00 sec  23.5 MBytes  19.7 Mbits/sec  0
[ 5] 0.00-10.00 sec  22.4 MBytes  18.8 Mbits/sec  0
sender
receiver
iperf Done.
```

(b) Client: M2-M1 connection

```
Server listening on 5201
Accepted connection from 117.211.211.44, port 60211
[ 5] local 192.168.1.29 port 5201 connected to 117.211.211.44 port 60212
[ ID] Interval      Transfer      Bandwidth
[ 5] 0.00-1.00 sec  2.80 MBytes  23.5 Mbits/sec
[ 5] 1.00-2.00 sec  3.61 MBytes  30.2 Mbits/sec
[ 5] 2.00-3.00 sec  3.88 MBytes  32.7 Mbits/sec
[ 5] 3.00-4.01 sec  3.76 MBytes  31.2 Mbits/sec
[ 5] 4.01-5.00 sec  3.81 MBytes  32.3 Mbits/sec
[ 5] 5.00-6.01 sec  3.80 MBytes  31.6 Mbits/sec
[ 5] 6.01-7.01 sec  3.82 MBytes  32.1 Mbits/sec
[ 5] 7.01-8.00 sec  3.67 MBytes  31.0 Mbits/sec
[ 5] 8.00-9.01 sec  3.90 MBytes  32.4 Mbits/sec
[ 5] 9.01-10.00 sec  3.82 MBytes  32.4 Mbits/sec
[ 5] 10.00-10.07 sec 259 KBytes  29.7 Mbits/sec
[ ID] Interval      Transfer      Bandwidth
[ 5] 0.00-10.07 sec  0.00 Bytes   0.00 bits/sec
[ 5] 0.00-10.07 sec  37.1 MBytes  30.9 Mbits/sec
sender
receiver
```

(c) Server: M2-M3 connection

```
SamBitts-MBP:~ lostsam423$ iperf3 -c 171.76.13.170
Connecting to host 171.76.13.170, port 5201
[ 5] local 192.168.1.3 port 60212 connected to 171.76.13.170 port 5201
[ ID] Interval      Transfer      Bitrate
[ 5] 0.00-1.00 sec  3.09 MBytes  25.9 Mbits/sec
[ 5] 1.00-2.00 sec  3.58 MBytes  30.1 Mbits/sec
[ 5] 2.00-3.00 sec  3.89 MBytes  32.6 Mbits/sec
[ 5] 3.00-4.00 sec  3.74 MBytes  31.3 Mbits/sec
[ 5] 4.00-5.00 sec  3.85 MBytes  32.3 Mbits/sec
[ 5] 5.00-6.00 sec  3.76 MBytes  31.5 Mbits/sec
[ 5] 6.00-7.00 sec  3.84 MBytes  32.3 Mbits/sec
[ 5] 7.00-8.00 sec  3.69 MBytes  31.0 Mbits/sec
[ 5] 8.00-9.00 sec  3.88 MBytes  32.5 Mbits/sec
[ 5] 9.00-10.00 sec  3.85 MBytes  32.3 Mbits/sec
[ ID] Interval      Transfer      Bitrate
[ 5] 0.00-10.00 sec  37.2 MBytes  31.2 Mbits/sec
[ 5] 0.00-10.00 sec  37.1 MBytes  31.2 Mbits/sec
sender
receiver
iperf Done.
SamBitts-MBP:~ lostsam423$
```

(d) Client: M2-M3 connection

Figure 6: Screenshots for TCP connection

```
iperf3: OUT OF ORDER - incoming packet = 235 and received packet = 3696 AND SP = 5
iperf3: OUT OF ORDER - incoming packet = 236 and received packet = 3720 AND SP = 5
iperf3: OUT OF ORDER - incoming packet = 237 and received packet = 3720 AND SP = 5
iperf3: OUT OF ORDER - incoming packet = 238 and received packet = 3750 AND SP = 5
iperf3: OUT OF ORDER - incoming packet = 244 and received packet = 3883 AND SP = 5
[ ID] Interval      Transfer      Bandwidth      Jitter      Lost/Total Datagrams
[ 5] 0.00-1.00 sec  6.68 MBytes  56.0 Mbits/sec  0.339 ms  29151/34059 (86%)
[ 5] 1.00-2.00 sec  11.4 MBytes  95.5 Mbits/sec  0.233 ms  38532/46989 (82%)
[ 5] 2.00-3.00 sec  11.4 MBytes  95.5 Mbits/sec  0.248 ms  37821/46278 (82%)
[ 5] 3.00-4.00 sec  11.4 MBytes  95.5 Mbits/sec  0.119 ms  36421/44878 (81%)
[ 5] 4.00-5.00 sec  11.4 MBytes  95.5 Mbits/sec  0.107 ms  38652/47109 (82%)
[ 5] 5.00-6.00 sec  11.4 MBytes  95.5 Mbits/sec  0.138 ms  37534/45992 (82%)
[ 5] 6.00-7.00 sec  11.4 MBytes  95.5 Mbits/sec  0.107 ms  38056/46513 (82%)
[ 5] 7.00-8.00 sec  11.4 MBytes  95.5 Mbits/sec  0.106 ms  36884/45341 (81%)
[ 5] 8.00-9.00 sec  11.4 MBytes  95.5 Mbits/sec  0.108 ms  39502/47959 (82%)
[ 5] 9.00-10.00 sec  11.4 MBytes  95.5 Mbits/sec  0.108 ms  37843/46301 (82%)
[ 5] 10.00-10.31 sec 1.97 MBytes  54.0 Mbits/sec  0.106 ms  6467/7928 (82%)
[ ID] Interval      Transfer      Bandwidth      Jitter      Lost/Total Datagrams
[ 5] 0.00-10.31 sec  0.00 Bytes   0.00 bits/sec  0.106 ms  376863/459347 (82%)
[SUM] 0.0-10.3 sec  50 datagrams received out-of-order
```

(a) Server: M2-M1 connection

```
root@docker-desktop:/# iperf3 -c 171.76.13.170 -u -b 0M
Connecting to host 171.76.13.170, port 5201
[ 5] local 192.168.65.3 port 57229 connected to 171.76.13.170 port 5201
[ ID] Interval      Transfer      Bitrate      Total Datagrams
[ 5] 0.00-1.00 sec  50.6 MBytes  491 Mbits/sec  43500
[ 5] 1.00-2.00 sec  62.5 MBytes  524 Mbits/sec  46300
[ 5] 2.00-3.00 sec  62.8 MBytes  527 Mbits/sec  46640
[ 5] 3.00-4.00 sec  61.7 MBytes  518 Mbits/sec  45830
[ 5] 4.00-5.00 sec  62.3 MBytes  523 Mbits/sec  46300
[ 5] 5.00-6.00 sec  63.1 MBytes  529 Mbits/sec  46850
[ 5] 6.00-7.00 sec  62.4 MBytes  523 Mbits/sec  46340
[ 5] 7.00-8.00 sec  61.6 MBytes  517 Mbits/sec  45770
[ 5] 8.00-9.00 sec  59.6 MBytes  500 Mbits/sec  44250
[ 5] 9.00-10.00 sec  58.8 MBytes  494 Mbits/sec  43700
[ ID] Interval      Transfer      Bitrate      Jitter      Lost/Total Datagrams
[ 5] 0.00-10.00 sec  613 MBytes  515 Mbits/sec  0.000 ms  0/455560 (0%) sender
[ 5] 0.00-10.00 sec  111 MBytes  92.9 Mbits/sec  0.096 ms  373424/455557 (82%) receiver
perf Done.
```

(b) Client: M2-M1 connection

```
Server listening on 5201
Accepted connection from 117.211.211.44, port 60223
[ 5] local 192.168.1.29 port 5201 connected to 117.211.211.44 port 58353
iperf3: OUT OF ORDER - incoming packet = 31 and received packet = 34 AND SP = 5
iperf3: OUT OF ORDER - incoming packet = 32 and received packet = 34 AND SP = 5
[ ID] Interval      Transfer      Bandwidth      Jitter      Lost/Total Datagrams
[ 5] 0.00-1.00 sec  8.83 MBytes  74.0 Mbits/sec  0.167 ms  173229/179781 (96%)
[ 5] 1.00-2.00 sec  11.4 MBytes  95.5 Mbits/sec  0.177 ms  237474/245930 (97%)
[ 5] 2.00-3.00 sec  11.4 MBytes  95.5 Mbits/sec  0.249 ms  230585/239042 (96%)
[ 5] 3.00-4.00 sec  11.4 MBytes  95.5 Mbits/sec  0.231 ms  243536/251994 (97%)
[ 5] 4.00-5.00 sec  11.4 MBytes  95.5 Mbits/sec  0.199 ms  237528/245985 (97%)
[ 5] 5.00-6.00 sec  10.7 MBytes  89.8 Mbits/sec  0.331 ms  253746/261693 (97%)
[ 5] 6.00-7.00 sec  10.1 MBytes  84.7 Mbits/sec  0.222 ms  243791/251290 (97%)
[ 5] 7.00-8.00 sec  10.1 MBytes  84.5 Mbits/sec  0.234 ms  238483/245965 (97%)
[ 5] 8.00-9.00 sec  9.13 MBytes  76.6 Mbits/sec  0.329 ms  240996/247778 (97%)
[ 5] 9.00-10.00 sec  7.03 MBytes  59.0 Mbits/sec  0.422 ms  241190/246413 (98%)
[ 5] 10.00-10.21 sec 1.50 MBytes  58.9 Mbits/sec  0.184 ms  54441/55554 (98%)
[ ID] Interval      Transfer      Bandwidth      Jitter      Lost/Total Datagrams
[ 5] 0.00-10.21 sec  0.00 Bytes   0.00 bits/sec  0.184 ms  2394999/2471425 (97%)
[SUM] 0.0-10.2 sec  2 datagrams received out-of-order
```

(c) Server: M2-M3 connection

```
SamBitts-MBP:~ lostsam423$ iperf3 -c 171.76.13.170 -u -b 0M
Connecting to host 171.76.13.170, port 5201
[ 5] local 192.168.1.3 port 58353 connected to 171.76.13.170 port 5201
[ ID] Interval      Transfer      Bitrate      Total Datagrams
[ 5] 0.00-1.00 sec  321 MBytes  2.69 Gbits/sec  238400
[ 5] 1.00-2.00 sec  336 MBytes  2.81 Gbits/sec  249180
[ 5] 2.00-3.00 sec  337 MBytes  2.83 Gbits/sec  250350
[ 5] 3.00-4.00 sec  336 MBytes  2.82 Gbits/sec  249630
[ 5] 4.00-5.00 sec  325 MBytes  2.72 Gbits/sec  241200
[ 5] 5.00-6.00 sec  337 MBytes  2.83 Gbits/sec  250120
[ 5] 6.00-7.00 sec  337 MBytes  2.83 Gbits/sec  250390
[ 5] 7.00-8.00 sec  337 MBytes  2.82 Gbits/sec  250070
[ 5] 8.00-9.00 sec  332 MBytes  2.79 Gbits/sec  246880
[ 5] 9.00-10.00 sec  333 MBytes  2.79 Gbits/sec  247190
[ ID] Interval      Transfer      Bitrate      Jitter      Lost/Total Datagrams
[ 5] 0.00-10.00 sec  3.25 GBytes  2.79 Gbits/sec  0.000 ms  0/2473410 (0%) sender
[ 5] 0.00-10.00 sec  103 MBytes  86.3 Mbits/sec  0.184 ms  2394999/2471425 (97%) receiver
iperf Done.
SamBitts-MBP:~ lostsam423$
```

(d) Client: M2-M3 connection

Figure 7: Screenshots for UDP connection

**Note:**

- (a) The website given for traceroute from different cities in the problem statement was not working. So we used <https://www.dotcom-tools.com/network-trace-test>.
- (b) Also, for Windows `tracert` wasn't available, so an analogous service `tracert` is used to collect data.
- (c) The measurements by M1 for CNN and IITD site were taken using an ubuntu container in Docker as it was not working on Mac terminal. Also, -I flag was used.
- (d) iperf measurements by M1 were also taken using ubuntu container.