Lab Exercise 3 (part B) Answers

Marking: Exercise 1: 1 mark, Exercise 2: 3 marks and Exercise 3: 6 marks.

Exercise 1: Investigate Performance of a Linear Topology

Question 1. (1 mark)

Source code provided on the lab webpage.

The output for the SimpleTest experiment is shown below. The ping test demonstrates that each host can communicate with every other host in the topology.

```
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (s2, s1) (s3, s2) (s4, s3)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
Dumping host connections
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
h4 h4-eth0:s4-eth1
Testing network connectivity
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
*** Stopping 1 controllers
c0
*** Stopping 7 links
.......
*** Stopping 4 switches
s1 s2 s3 s4
*** Stopping 4 hosts
h1 h2 h3 h4
*** Done
```

Question 2.

The output for the PerformanceTest experiment is shown below. In addition to the ping test (as above), the iperf tool is run to test the TCP throughput between hosts h1 and h4, which is reported to be 11.3 Gbits/sec.

```
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3 s4
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (s2, s1) (s3, s2) (s4, s3)
```

```
*** Configuring hosts
h1 (cfs -1/100000us) h2 (cfs -1/100000us) h3 (cfs -1/100000us) h4 (cfs -
1/100000us)
*** Starting controller
c0
*** Starting 4 switches
s1 s2 s3 s4 ...
Dumping host connections
h1 h1-eth0:s1-eth1
h2 h2-eth0:s2-eth1
h3 h3-eth0:s3-eth1
h4 h4-eth0:s4-eth1
Testing network connectivity
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
Testing bandwidth between h1 and h4
*** Iperf: testing TCP bandwidth between h1 and h4
*** Results: ['11.3 Gbits/sec', '11.3 Gbits/sec']
*** Stopping 1 controllers
c0
*** Stopping 7 links
.......
*** Stopping 4 switches
s1 s2 s3 s4
*** Stopping 4 hosts
h1 h2 h3 h4
*** Done
```

Question 3.

I ran the server on host h4 and the client on host h1. The following is the output of the iperf tool, which indicates that the TCP throughput (i.e. bandwidth) between the two hosts is 20.2 Gbits/sec.

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 35728 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.0 sec  23.6 GBytes  20.3 Gbits/sec
```

Note: It is not uncommon to observe slightly different measurements at the client and server. T

Question 4.

The *–r* flag can be used to conduct a bidirectional test sequentially. The following is the output which indicates that the throughput in the forward direction is 17.3 Gbits/sec and the reverse direction is 22.4 Gbits/sec.

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -r
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 25] local 10.0.0.1 port 35731 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 25]  0.0-10.0 sec  20.1 GBytes  17.3 Gbits/sec
[ 24] local 10.0.0.1 port 5001 connected with 10.0.0.4 port 44735
[ 24]  0.0-10.0 sec  26.1 GBytes  22.4 Gbits/sec
```

Question 5.

The *–d* flag can be used to conduct a bidirectional test simultaneously. The following is the output which indicates that the throughput in the forward direction is 8.33 Gbits/sec and the reverse direction is 10.7 Gbits/sec. Observe that the throughput for each direction is approximately 50% of the result for the sequential experiment (reported in Question 4). This may seem strange since typically all network links are full-duplex and have the same capacity in each direction. However, in the linear topology we have not specified any bandwidth for the links. As such, mininet will use the full capacity of the host machine which appears to be around 20-22 Gbits/sec for the machine that I was using for the tests. As such, when we run two parallel TCP connections, we observe this drop in throughput for each connection (since the max total traffic at any given time is limited to around 20-22 Gbits/sec). You will observe a different behavior in Experiment 2.

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -d
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 35729 connected with 10.0.0.4 port 5001
[ 25] local 10.0.0.1 port 5001 connected with 10.0.0.4 port 44733
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.0 sec  9.70 GBytes  8.33 Gbits/sec
[ 25]  0.0-10.0 sec  12.4 GBytes  10.7 Gbits/sec
```

Question 6.

The –u flag should be used (both at server and client) to conduct bandwidth measurement using UDP. The following is the output:

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -u
------------------------------------------------------------
Client connecting to 10.0.0.4, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 43544 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec
[ 23] Sent 893 datagrams
[ 23] Server Report:
[ 23]  0.0-10.0 sec  1.25 MBytes  1.05 Mbits/sec   0.022 ms    0/  893 (0%)
```

Note that in the case of UDP measurements, iperf will use a default bandwidth of 1 Mbps for the links. We can change the bandwidth by using the –b option. See below for an example:

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -u -b100m
------------------------------------------------------------
Client connecting to 10.0.0.4, UDP port 5001
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 58987 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.0 sec   120 MBytes   101 Mbits/sec
[ 23] Sent 85460 datagrams
[ 23] Server Report:
[ 23]  0.0-10.0 sec   120 MBytes   101 Mbits/sec   0.006 ms    0/85460 (0%)
```

Exercise 2: Learning to use network configuration

Question 7 (3 marks)

As before, I ran the iperf server on h4 and iperf client on h1.

The output this time around for the basic TCP test is as follows. Observe that the measured throughput is much lower, 1.92Mbits/sec, as compared to around 20Gbits/sec in Exercise 1. Moreover, the throughput is significantly lower than the bandwidth of the links (10 Mbits/sec).

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 35861 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-12.0 sec  2.75 MBytes  1.92 Mbits/sec
```

The output for the bidirectional sequential experiment is below

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -r
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 35862 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.5 sec  2.88 MBytes  2.30 Mbits/sec
[ 25] local 10.0.0.1 port 5001 connected with 10.0.0.4 port 44866
[ 25]  0.0-10.7 sec  3.50 MBytes  2.74 Mbits/sec
```

And the output for the bidirectional but simultaneous experiment is below. Observe that unlike Experiment 1 (Question 5), the measured throughput in each direction is similar to that for the sequential measurements (above).

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -d
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
------------------------------------------------------------
Client connecting to 10.0.0.4, TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 35864 connected with 10.0.0.4 port 5001
[ 25] local 10.0.0.1 port 5001 connected with 10.0.0.4 port 44868
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.8 sec  3.12 MBytes  2.43 Mbits/sec
[ 25]  0.0-11.7 sec  3.38 MBytes  2.42 Mbits/sec
```

For the UDP test I configured the maximum bandwidth limit to a very large value (-b100m) so that I can measure the actual bandwidth of the custom topology. As indicated in the output below, the measured UDP throughput was 9.52Mbits/sec which is very close to the capacity of the links in the network (10Mbits/sec). Also observe that 3 datagrams were received out of order and only 89% of the datagrams send reached the destination. This is not uncommon with UDP tests.

```
root@mininet-vm:~/cs3331# iperf -c 10.0.0.4 -u -b100m
------------------------------------------------------------
Client connecting to 10.0.0.4, UDP port 5001
```

```
Sending 1470 byte datagrams
UDP buffer size:  208 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.1 port 51265 connected with 10.0.0.4 port 5001
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0-10.0 sec   120 MBytes   100 Mbits/sec
[ 23] Sent 85435 datagrams
[ 23] Server Report:
[ 23]   0.0-11.2 sec   12.7 MBytes   9.52 Mbits/sec    0.200 ms 76368/85434
(89%)
[ 23]  0.0-11.2 sec  3 datagrams received out-of-order
```

Exercise 3: Performance Evaluation for a Larger Topology

Question 8 (2 marks)

I have not included the text files here as this is straightforward. I only show the summary results that I obtained below:

L1 (measured between h1 and h2)
Throughput: 20.7 Mbits/sec
RTT: min: 80.389 ms, avg: 80.915 ms, max: 83.572 ms, mdev: 0.812 ms

L2 (measured between h2 and h3)
Throughput: 39 Mbits/sec
RTT: min: 20.2 ms, avg: 20.68 ms, max: 21.491 ms, mdev: 0.336 ms

L3 (measured between h3 and h4)
Throughput: 20.7 Mbits/sec
RTT: min: 80.394ms, avg: 80.955 ms, max: 83.373 ms, mdev: 0.707 ms

L4 (measured between h2 and h5)
Throughput: 20.1 Mbits/sec
RTT: min: 10.214ms, avg: 10.833 ms, max: 13.087 ms, mdev: 0.599 ms

L5 (measured between h3 and h6)
Throughput: 21.7 Mbits/sec
RTT: min: 10.247 ms, avg: 10.779 ms, max: 12.908 ms, mdev: 0.624 ms

The measured RTT and throughput are consistent with the link settins in the topology. For example, L1 is defined to have 20 Mbits/sec bandwidth and a one-way latency of 40ms. The measured throughput for L1 is approximately around 20 Mbits/sec and the RTT is around 80ms. Similar observations for all other links.

Question 9. (1 mark)

The summary results for the path from h1 to h4 are below:
Throughput: 19.2 Mbits/sec
RTT: min: 180.751 ms, avg: 182.434 ms, max The measured RTT and throughput are consistent with the topology settings.
: 187.8 ms, mdev: 1.549 ms

The path from h1 to h4 includes links L1, L2 and L3. The bottleneck link along this path are L1 and L3 (since both have the same capacity, 20 Mbits/sec). As such, the end to end throughput is around 20 Mbits/sec.

The RTT can be obtained by adding up the RTTs along L1 (80ms), L2 (20ms) and L3 (80ms), which equals 180 msec. The measured RTT is close to this value.

Question 10. (2 marks)

I conducted the measurements between the following host pairs: (i) h1 -> h4 and (ii) h7 -> h9. Here are the summary results:

h1 –> h4
Throughput: 9.92 Mbits/sec
RTT: min: 181.298 ms, avg: 182.678 ms, max: 187.007 ms, mdev: 1.423 ms

h7 –> h9
Throughput: 11.12 Mbits/sec
RTT: min: 181.506 ms, avg: 182.9 ms, max: 188.741 ms, mdev: 1.624 ms

Observe that the RTT for both communicating pairs is around 180ms, which is consistent with the result for Question 9. The logic is the same. The RTT of the path is the summation of the RTT of each link along the path. The number of simultaneously communicating pairs does not significantly impact the RTT since TCP uses congestion avoidance and thus router queues do not build up and queueing delays are insignificant.

As noted in Question 9, the end-to-end throughput of the path between S1 and S4 is 20Mbit/sec. We can observe that the sum of the TCP throughput for the two communicating pairs adds up to approximately this value. However, observe that the capacity is not exactly equally shared. This may be due to the short duration of the experiment (20 seconds).

Next I choose the following 3 host pairs: (i) h1 -> h4 and (ii) h7 -> h9 and (iii) h8 –> h10. The results summary is below:

h1 –> h4
Throughput: 7.53 Mbits/sec
RTT: min: 180.910 ms, avg: 182.604 ms, max: 187.001 ms, mdev: 1.332 ms

h7 –> h9
Throughput: 5.64 Mbits/sec
RTT: min: 181.150 ms, avg: 182.678 ms, max: 187.413 ms, mdev: 1.341 ms

h8 –> h10
Throughput: 7.47 Mbits/sec
RTT: min: 181.030 ms, avg: 182.226 ms, max: 187.787 ms, mdev: 1.527 ms

Notice that the RTT for the 3 communicating pairs is again around 180ms, which is consistent with the previous observations. As above, the sum of the throughput for the 3 pairs is approximately around 20 Mbits/sec.

Question 11. (1 mark)

The average results are summarized below:

h1 –> h4
Throughput: 21.3 Mbits/sec
RTT: min: 181.479 ms, avg: 182.687 ms, max: 187.628 ms, mdev: 1.366 ms

h5-> h6
Throughput: 20.4 Mbits/sec
RTT: min: 41.157 ms, avg: 42.277 ms, max: 48.403 ms, mdev: 1.836 ms

The path from h1 to h4 spans L1, L2 and L3. The path from h5 to h6 spans L4, L2 and L5. As in the above questions, the measured RTT results are consistent with the delay settings for these links.

Observe that the two flows overlap on link L2 which has bandwidth of 40Mbits/sec. Assuming that this bandwidth is equally shared each flow should be able to transmit at 20 Mbits/sec on L2.

Since L1 and L3 also have a capacity of 20Mbits/sec, the throughput from h1->h4 should approximately be equal to 20Mbits/sec. Similarly, since L4 and L5 also have a capacity of 20Mbits/sec, the throughput from h5->h6 should also be approximately equal to 20 Mbits/sec. This is consistent with the measured throughput noted above.

Exercise 4: Performance Evaluation for a Larger Topology

Question 12.

Code uploaded separately.

Question 13.

I have not included the text files here as this is straightforward. I only show the summary results that I obtained below:

L1 (measured between h2 and h3)
Throughput: 21.3 Mbits/sec
RTT: min: 20.345 ms, avg: 21.011 ms, max: 24.813 ms, mdev: 1.009 ms

L2 (measured between h3 and h4)
Throughput: 11.7 Mbits/sec
RTT: min: 30.152 ms, avg: 30.708 ms, max: 33.158 ms, mdev: 0.708 ms

L3 (measured between h4 and h5)
Throughput: 21.2 Mbits/sec
RTT: min: 40.215 ms, avg: 40.782 ms, max: 42.798 ms, mdev: 0.623 ms

The measured RTT and throughput are consistent with the topology settings. (as in Question 8).

Question 14.

I have not included the text files here as this is straightforward. I only show the summary results that I obtained below:

Throughput: 11.5 Mbits/sec
RTT: min: 90.810 ms, avg: 92.168 ms, max: 98.043 ms, mdev: 1.912 ms

The path from h1 to h5 spans L1, L2 and L3. The bottleneck link along this path is L2, which has bandwidth of 10 Mbits/sec. Thus, the expected end-to-end throughput for this connection will be around 10 Mbits/sec. This is consistent with the measured throughput.

Adding up the round-trip latencies along these links, gives us, 20 + 30 + 40 = 90 ms. This is consistent with the measured RTT.

Question 15.

I conducted the measurements between the following host pairs: (i) h1 -> h5 and (ii) h2 -> h6. Here are the summary results:

h1->h5
Throughput: 6.15 Mbits/sec
RTT: min: 90.819 ms, avg: 92 ms, max: 98.259 ms, mdev: 1.684 ms

h2->h6
Throughput: 4.98 Mbits/sec
RTT: min: 91.115 ms, avg: 92.372 ms, max: 97.580 ms, mdev: 1.442 ms

The RTT measurements are consistent with the RTT for this path as was measured in Question 14. The sum of the throughput for the two flows adds up to around 10 Mbits/sec, which is also consistent with the capacity of this end-to-end path as noted in Question 14.

Question 16.

h1->h6
Throughput: 6.05 Mbits/sec
RTT: min: 91.199 ms, avg: 92.245 ms, max: 96.023 ms, mdev: 1.188 ms

h3 -> h4
Throughput: 5.18 Mbits/sec
RTT: min: 30.242 ms, avg: 30.881 ms, max: 34.252 ms, mdev: 0.967 ms

The path from h1 to h6 spans L1, L2 and L3. As such, the RTT measurements are consistent wit the RTT for this past as was measured in Question 15.

The path from h3 to h4 spans L2 which has a one-way latency of 15 ms. The measured RTT is thus consistent with the link delay.

The two flows under consideration overlap on link L2. As such, the link capacity for L2 (10 Mbits/sec) will be approximately equally shared between these two flows. This is reflected in the measured throughput for h3->h4.

L2 is also the bottleneck link for h1->h6 and as such the measured throughput for this flow is also around 5Mbits/sec.