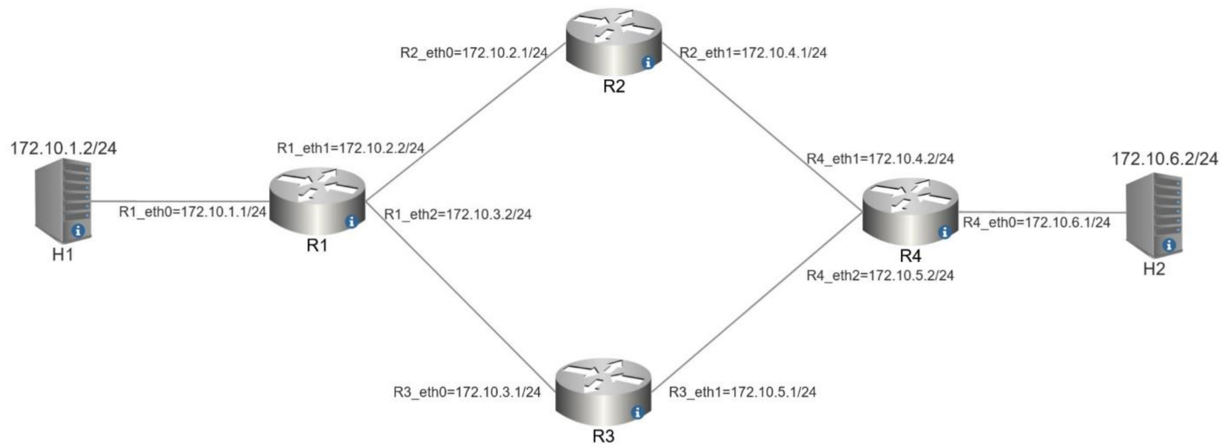

Figure 1: Network Topology

C1:

I created the above topology on mininet using the mininet Python API. It consists of 2 hosts, H1 and H2, connected through a network of 4 routers R1-R4.

In this part, we use these below commands to limit router bandwidth, buffer size, and introduce delays in every interface of all routers with the tc tbf and tc netem options respectively.

tc qdisc add dev <interface> root handle 1: tbf rate <bandwidth> burst <burst_rate> limit <buffer_size>
tc qdisc add dev <interface> parent 1:1 handle 10: netem delay <delay_in_ms>

BIRD is used to run the RIP protocol for dynamic routing. Once the routers have been reconfigured with the above commands, we then spawn xterm instances of hosts H1 and H2 to run the network performance tool, Iperf, which will simulate a TCP performance test between the two hosts, with H2 acting as the Iperf server and H1 the client. We do 3 measurements, with the router buffer size varying as 10Kb, 5Mb and 25Mb across them. Delay = 30ms and Bandwidth = 100Mbps are kept constant in all 3 runs.

To run H2 as the Iperf server, we use the below command on its xterm terminal
iperf3 -s
To run H1 as the Iperf client, we use the below command on its xterm terminal
iperf3 -c <H2_IP>

To create a json output file we can use -J option and pipe the result of both server and client.

We first run the MyIperf.py file in the partC directory that automates the configuration of the routers with the tc commands. To verify that the routers have been configured correctly we can use these two command that is shown in Figure 2.

```
mininet> R1 tc qdisc show
qdisc noqueue 0: dev lo root refcnt 2
qdisc tbf 1: dev r1-eth0 root refcnt 2 rate 100Mbit burst 10Kb lat 0us
qdisc netem 10: dev r1-eth0 parent 1:1 limit 1000 delay 30.0ms
qdisc tbf 1: dev r1-eth1 root refcnt 2 rate 100Mbit burst 10Kb lat 0us
qdisc netem 10: dev r1-eth1 parent 1:1 limit 1000 delay 30.0ms
qdisc tbf 1: dev r1-eth2 root refcnt 2 rate 100Mbit burst 10Kb lat 0us
qdisc netem 10: dev r1-eth2 parent 1:1 limit 1000 delay 30.0ms
mininet> H1 ping -c 10 H2
PING 172.10.6.2 (172.10.6.2) 56(84) bytes of data.
64 bytes from 172.10.6.2: icmp_seq=1 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=2 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=3 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=4 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=5 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=6 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=7 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=8 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=9 ttl=61 time=181 ms
64 bytes from 172.10.6.2: icmp_seq=10 ttl=61 time=181 ms

--- 172.10.6.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9007ms
rtt min/avg/max/mdev = 181.493/181.632/181.978/0.445 ms
```

Figure 2

In the Figure 2, we can see that all interfaces of router R1 have been correctly configured by the tc command to update its bandwidth, buffer_size and delay parameters. Similarly, we can verify the same for the other routers as well.

Also from the ping command, we see that the RTT delay is now 180ms. Every packet flow through 6 interfaces from H1 to H2 (refer topology diagram in Figure 1, the interfaces are H1 -> r1-eth0 -> r1-eth1 -> r2-eth0 -> r2-eth1 -> r4-eth1 -> r4-eth0 -> H2), and hence total delay is 6 * 30 = 180ms per packet.

Now that we have setup the routers, we will run iperf between H1 and H2 through their xterms, as shown below. We will then describe the results for buffer_size = 10Kb, 5Mb and 25Mb.

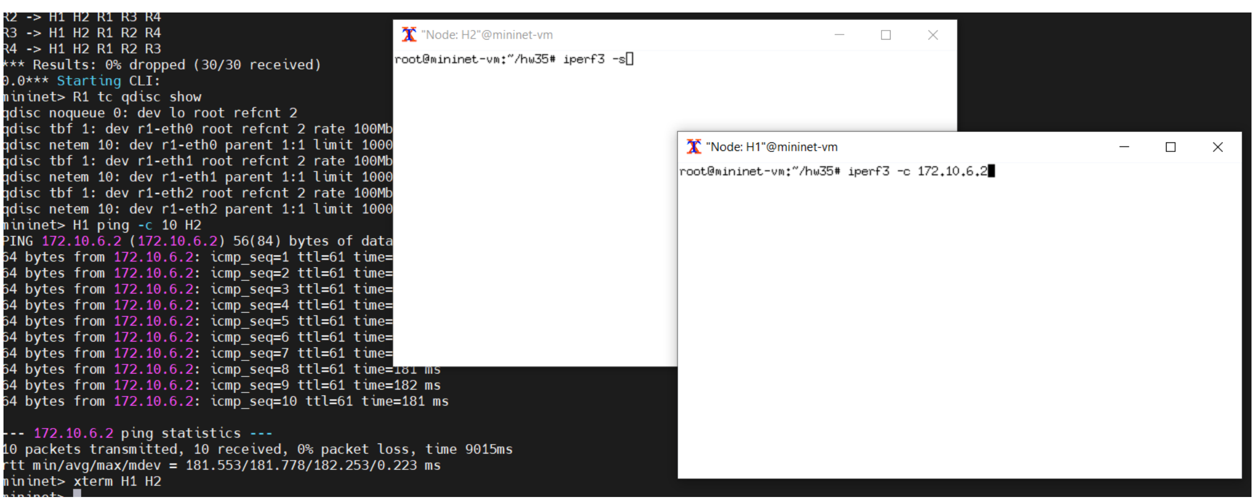

Figure 3

We also capture the client/server result json files for each buffer_size configuration and store them in the *partC/iperf_results* directory.

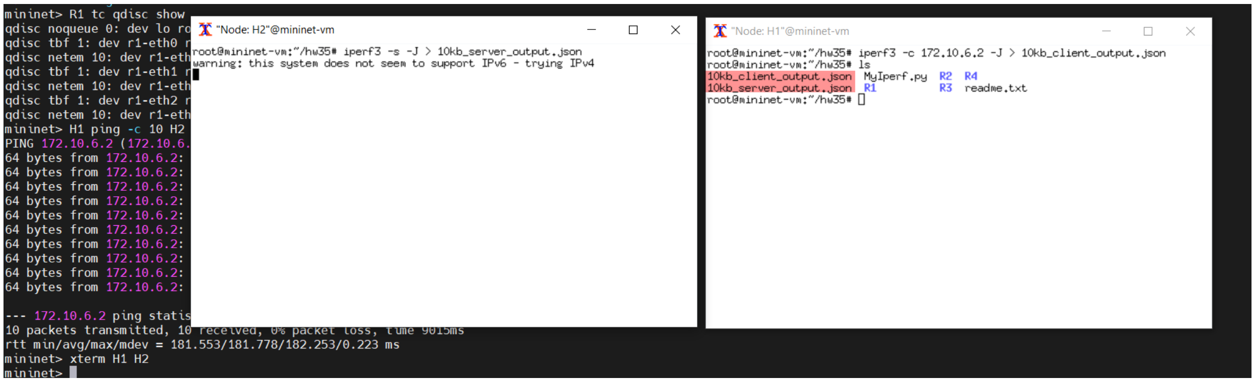

Figure 4

To estimate the actual bandwidth, number of retransmissions and the round-trip times of the entire flow from the '*bits_per_second*', '*retransmits*' and '*mean_rtt*' attributes (in *$.end.streams[0].sender.** JSONPath) in the *_client_output.json* files, as shown in Figure 5. The Bandwidth Delay Product (BDP) can then be calculated as BDP = bits_per_second * mean_rtt

Figure 5

Test1. Buffer Size 10Kb

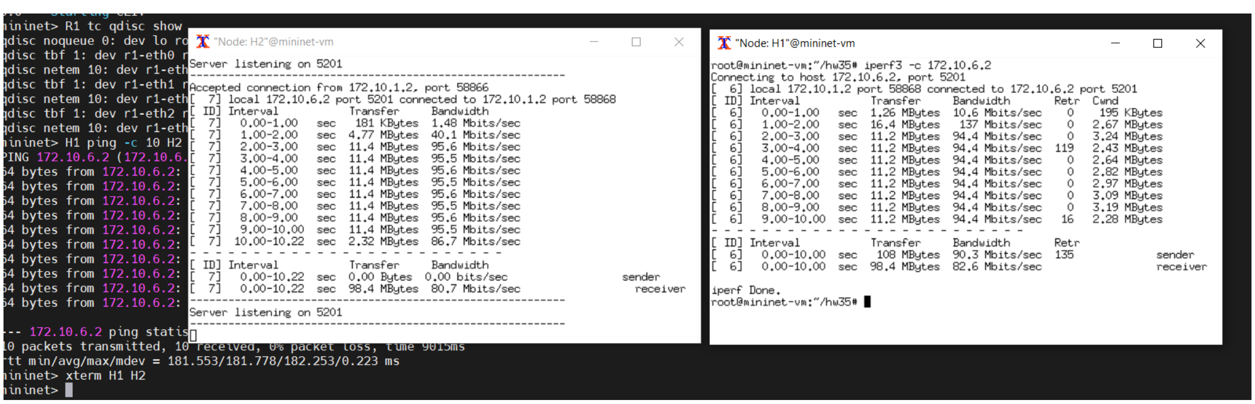

Figure 6

BDP = 89226774.540034 * 248779 * 10^{-6} = 2.10 Mbytes

Test2. Buffer Size 5mb

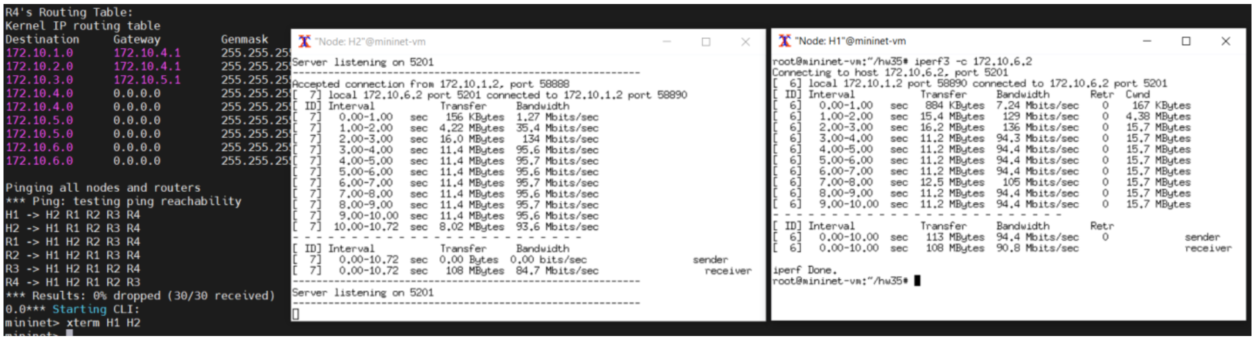

Figure 7

BDP = 94794209.244125 * 595674 * 10^{-6} = 53.85 Mbytes

Test3. Buffer Size 25mb

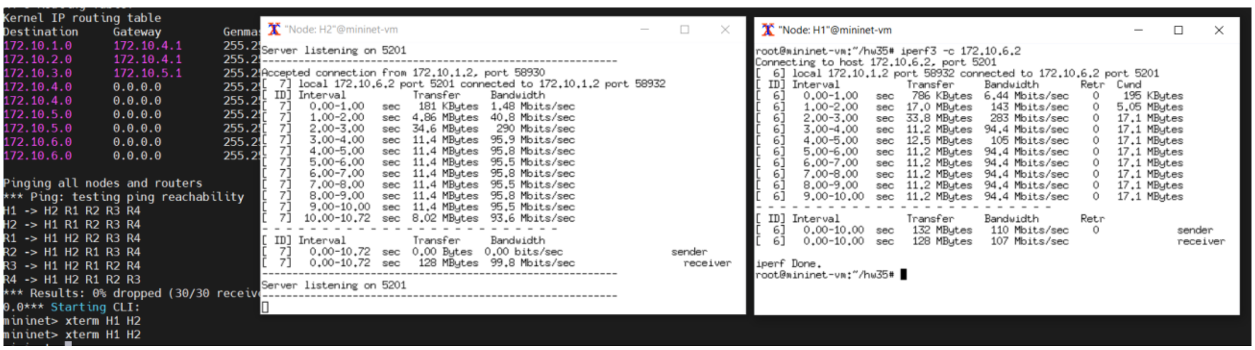

Figure 8

BDP = 110724500.982998 * 544719 * 10^{-6} = 57.51 Mbytes

OBERVATIONS:

1. We see some retransmissions in the first measurement with 10Kb buffer size. This may be due to packets being dropped from one of the router interfaces due to the small-size buffer queue filling up. For the other 2 measurements, we see that there are no retransmissions after the router buffer size was increased.

2. The BDP is increasing with increase in buffer_size. This is because of increase in the actual average bandwidth (refer Bitrate in H1) with increase in buffer_size.

3. We see a major multiplicative increase of the cwnd in the first 2 seconds, followed by a sharp drop in the bandwidth (bitrate) which then leads to the cwnd stabilizing across successive intervals.

4. Since retransmission does not occur for buffer_sizes 5Mb and 25Mb, the retransmission time is 0. For 10Kb, the retransmission time may be around 0.24s (10.24s – 10s)