Problem1

500
completion time 28387 usec, bytes transferred 2898266, reliable throughput 102 mbps

1000
completion time 27394 usec, bytes transferred 2898266, reliable throughput 105 mbps

1500

completion time 26616 usec, bytes transferred 2898266, reliable throughput 108 mbps

2500
completion time 27554 usec, bytes transferred 2898266, reliable throughput 105 mbps

3500
completion time 27202 usec, bytes transferred 2898266, reliable throughput 106 mbps

4500
completion time 28242 usec, bytes transferred 2898266, reliable throughput 102 mbps

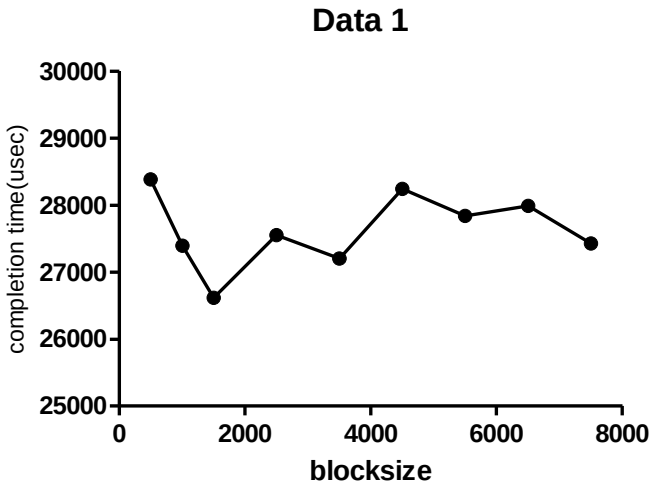
5500

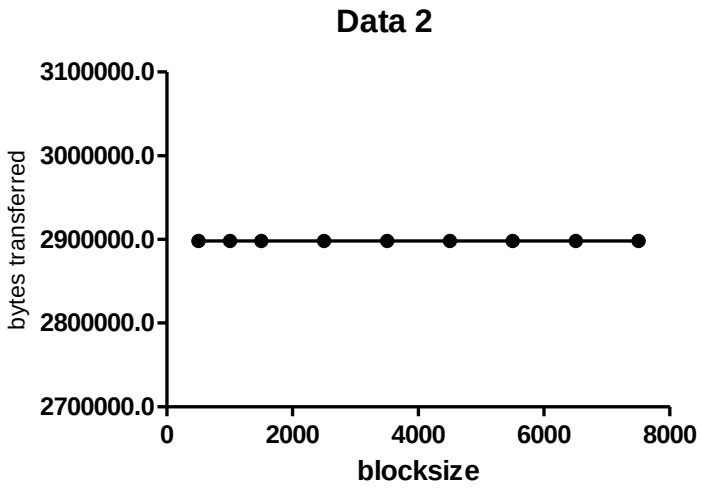
completion time 27839 usec, bytes transferred 2898266, reliable throughput 104 mbps

6500
completion time 27989 usec, bytes transferred 2898266, reliable throughput 103 mbps

7500
completion time 27429 usec, bytes transferred 2898266, reliable throughput 105 mbps

We can see that the client and server is around the same value.


Data 1
30000
29000
28000
27000
26000
25000
completion time(usec)
0
2000
4000
6000
8000
blocksize


Data 2
3100000.0
3000000.0
2900000.0
2800000.0
2700000.0
bytes transferred
0
2000
4000
6000
8000
blocksize

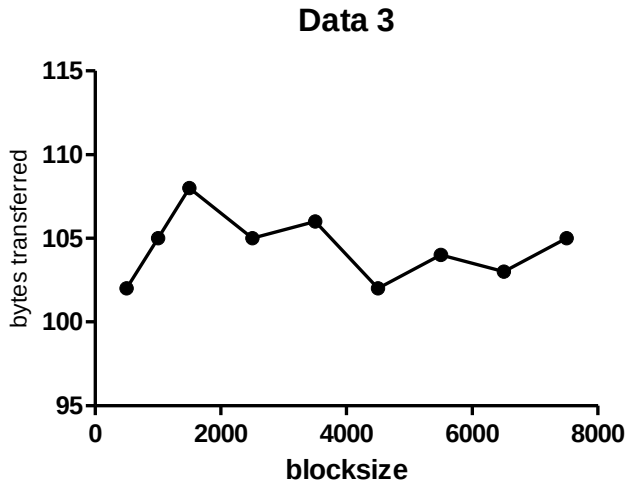

Data 3
115
110
105
100
95
bytes transferred
0
2000
4000
6000
8000
blocksize

Figure 1

We can see from the above Figure 1 that increasing blocksize does not always increase the bps. The optimal block size if around 1500.

Problem 2

./traffic_snd 128.10.3.60 59434 1000 1000 50

Completion time 50.104000,reliable bps 20876.576720, pps 19.958486.

Completion time 50.105000,reliable bps 20876.160064, pps 19.958088.

10

Completion time 10.103000,reliable bps 103533.603880, pps 98.980501.

Completion time 10.103000,reliable bps 103533.603880, pps 98.980501.

5

Completion time 5.104000,reliable bps 204937.304075, pps 195.924765.
Completion time 5.104000,reliable bps 204937.304075, pps 195.924765.

1

Completion time 1.094000,reliable bps 956124.314442, pps 914.076782.

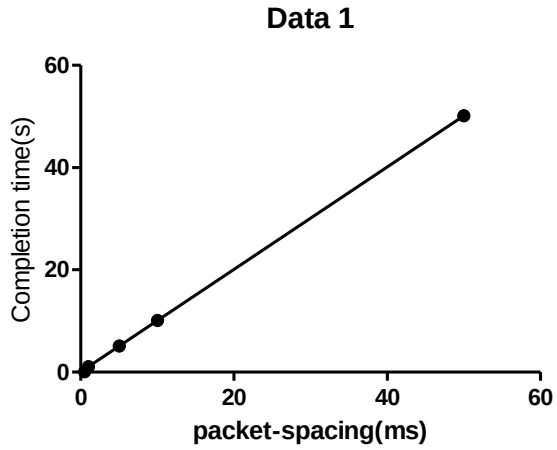
Completion time 1.094000,reliable bps 956124.314442, pps 914.076782.

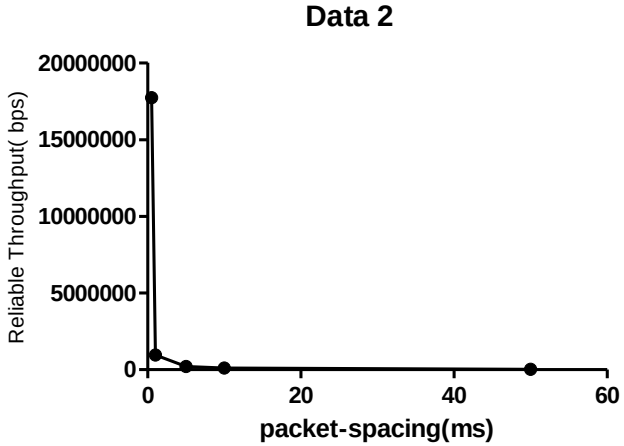
0.5

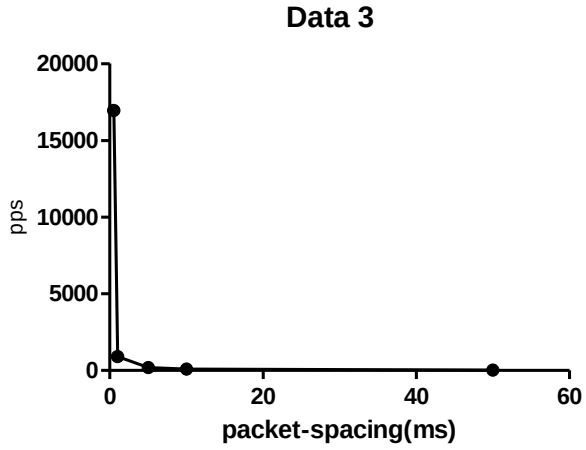
Completion time 0.059000,reliable bps 17728813.559322, pps 16949.152542.

Completion time 0.059000,reliable bps 17728813.559322, pps 16949.152542.

We can see that the client and server is around the same value.







We can see that the packet spacing greatly affects the reliable throughput. If the spacing increase to 50. The throughput will decrease greatly.

./traffic_snd 128.10.3.59 50101 50 10000 1

Completion time 10.960000,reliable bps 87591.240876, pps 912.408759.
Completion time 10.959000,reliable bps 87599.233507, pps 912.492016.

500
Completion time 10.965000,reliable bps 497948.016416, pps 911.992704.

Completion time 10.965000,reliable bps 497948.016416, pps 911.992704.

1000
Completion time 10.963000,reliable bps 954118.398249, pps 912.159081.
Completion time 10.963000,reliable bps 954118.398249, pps 912.159081.

2000
Completion time 11.010000,reliable bps 1858310.626703, pps 908.265213.
Completion time 11.010000,reliable bps 1858310.626703, pps 908.265213.

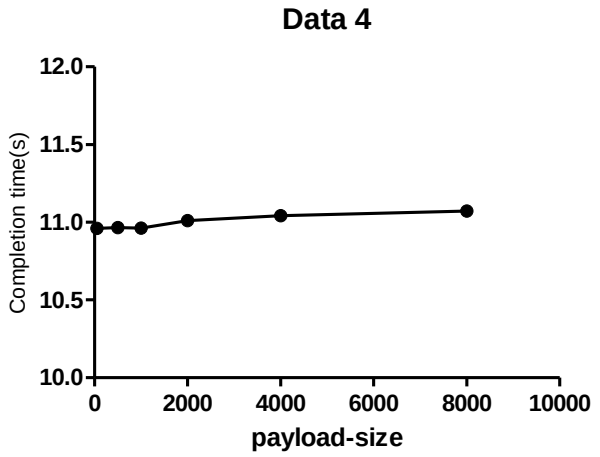
4000

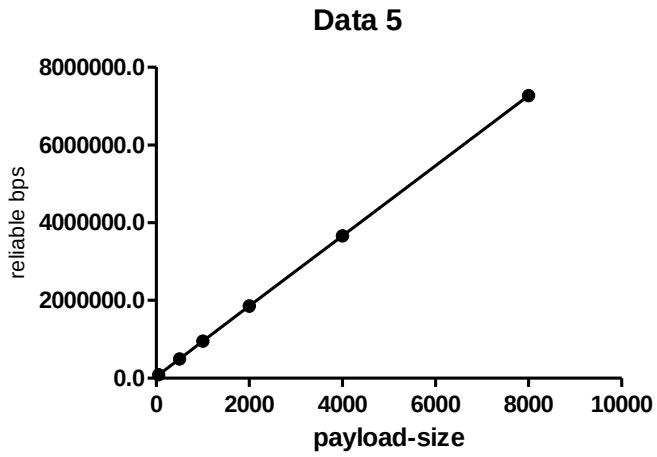
Completion time 11.043000,reliable bps 3663859.458480, pps 905.551028.
Completion time 11.043000,reliable bps 3663859.458480, pps 905.551028.

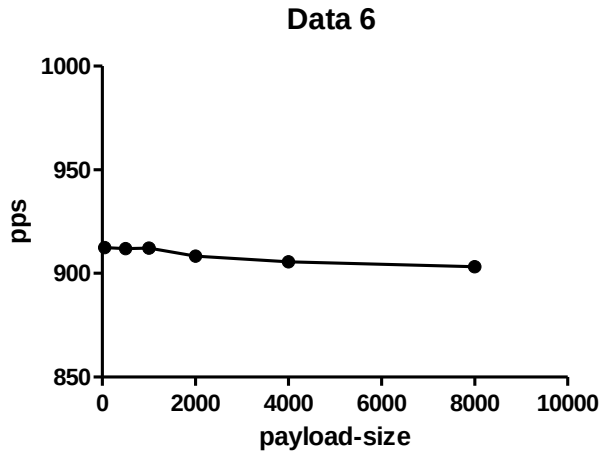
8000
Completion time 11.072000,reliable bps 7266979.768786, pps 903.179191.

Completion time 11.072000,reliable bps 7266979.768786, pps 903.179191.

We can see that the client and server is around the same value.







As we can see from the above figure. The payload size does not affect the completion time and pps a lot. The reliable pps is proportion to the payload size.

3.
First, based on the DIX and IEEE 802.3 Ethernet frame format shown in class, determine the 48-bit source and destination MAC addresses.

Src: 5e:5e:18:fa:9e:44 (5e:5e:18:fa:9e:44)
Dst: 96:48:e4:15:cc:ff (96:48:e4:15:cc:ff)

Second, inspect the next two bytes which may be a type (DIX) or length (802.3) field depending on its content. Find out what values for the DIX type field are defined which, with the limit placed on Ethernet payload size, will allow you to determine if the captured frames are DIX or 802.3.

0x0800 means it contains an IPv4 datagram.
Thus, it is DIX frame.

Third, the first 4 bits of an IP packet header contain its version number. IPv4 packet headers, by default, are 20 bytes in length. Inspect the first four bits and determine what the version number is. Keep in mind that the payload of Ethernet, in our setting, is an IP packet.

It is IPv4

Fourth, UDP packet headers are 8 bytes long, with the first 4 bytes specifying and source and destination ports. Keeping in mind a UDP packet is the payload of an IP packet for the traffic generation app, peek further inside and see if you can discern the source and destination port numbers.

Src Port: 39452, Dst Port: 50002

Finally, the payload of UDP is filled with the first letter of your last name by traffic_snd. Perform deep inspection to see if you can decode the payload of UDP. Describe your findings and how you went about determining the field values in lab3ans.pdf.

I used " tcpdump -qns 0 -A -r mylogfile" to check the payload of "L"s. I found that, between two successfully UDP transmission, there is one ICMP transmission of length 556.