Analysis

We have run the experiment by doing simulation of TCP/UDP performance by using Iperf command line tool to experiment with various TCP window size. For the first task, we have measured the number of packet which is sent before receive an acknowledgement (window size) and also see how much memory is allocated to send and receive traffic flow buffers (buffer length). The result can be illustrated by Table 1 in Appendix

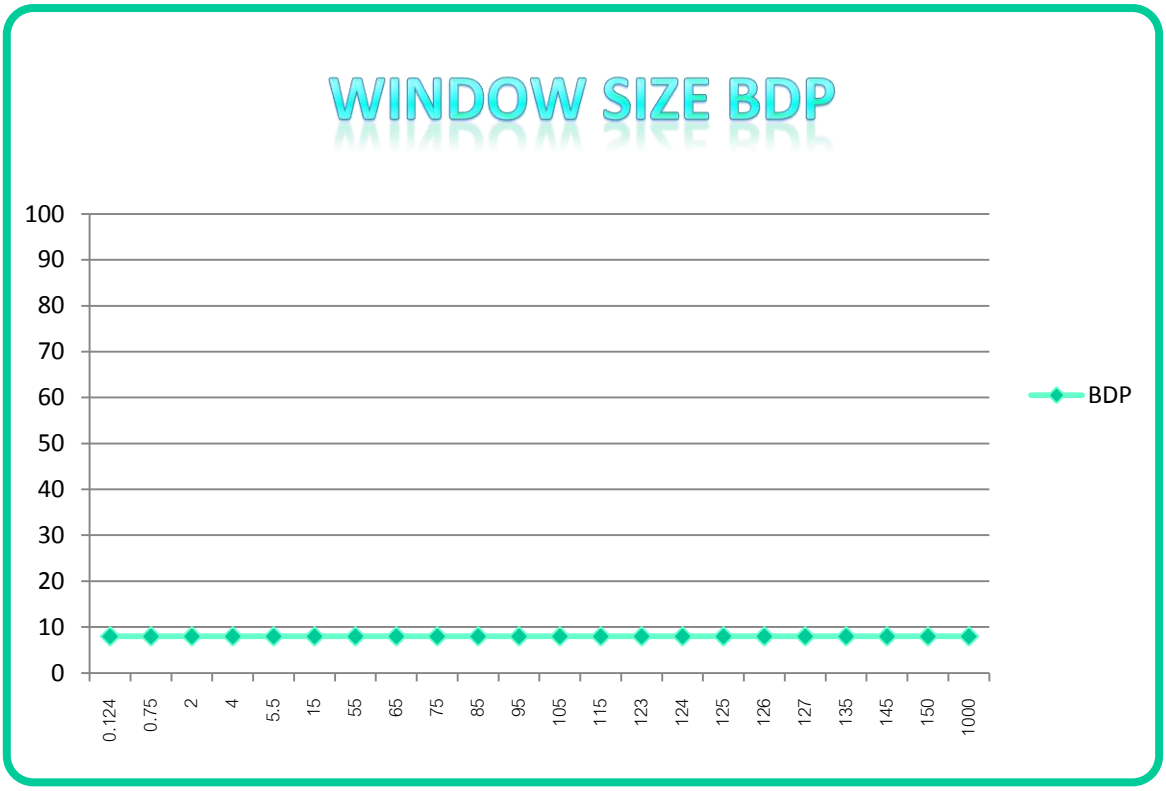


Figure 1:

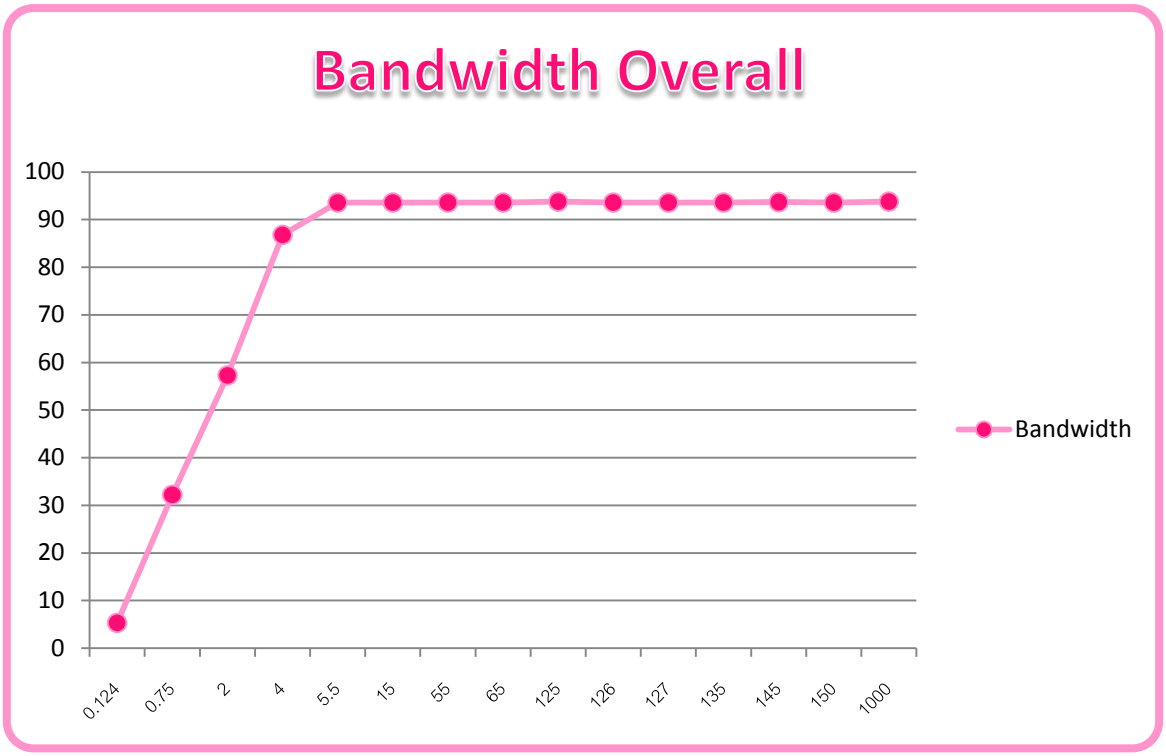


Figure 2:

Analysis 1.1)

As you can see from Figure 1, the performance at starting is low since the receive window that the receiver has been advertised is too low. However, if we increase window size until it reaches to 5.5 KB, the bandwidth will be stable and peaking at 93.8 Mbits/sec at the receive window is 125 KB.

For this scenario, the receiving TCP/client tries to allocate buffer size and use the window size to advertise the available buffer size to the sender/server. Briefly, we can say that the window size refers to receiver's buffer size.

Since the buffer is unlimited which we can explicitly see from table1 that the range of the window size value which given the maximum bandwidth of this experience is from 5.5 KB to indefinite value. We will see that the range is too large. The reason is since this experiment, we have got unlimited buffer which can be illustrated in Table1 in appendix.

To get full TCP performance the TCP window needs to be large enough to accommodate the Bandwidth Delay Product which is the product of bottleneck link bandwidth and the round trip time by using the formula as following:

RTT which obtain from ping command is: 0.637 ms

BDP = data rate *RTT

RTT can be obtained by using ping command.

BDP = 100x10^6 X 0.637

=7.9625 KB

By Definition, Bandwidth delay product (BDP) which describes the amount of buffering is required in the sending and receiving host. By default, the largest buffer is 64 KB. According to Table 1 in appendix , we can explicitly find that BDP is approximately 7.96 which is less than the default value. Therefore, the buffer is large enough. That mean, the buffer is unlimited.

We can obviously see that increasing the window size may not produce increasing throughput. Since we will see that even the receiver tries to advertise the window size is set to be larger than 125 KB, the throughput still nearly stable at 93.8 Mb/s.

Moreover, in the case of small receive window, the result is explicitly showed that the throughput is limited that since the receiver advertises only small receive window size which tell that the sender should be waited and shouldn't send more than that value, therefore, the throughput get lower than the high receive window.

We will see that at the maximum bandwidth, TCP can get high throughput. The reason is since TCP is reliable protocol. If there is some packet loss occur either from three Duplicate ACK or timeout events. TCP will always minimize retransmit the packet loss.

We can explicitly see that sometimes, the congestion window or sending rate will be reduced. That reduction from the transport link may not be stable which may give us worse performance.

Moreover, packet loss or bottleneck in the network may be one of the factors which lead to TCP throughput reduction.

To conclude, bandwidth delay product is the factor which impacts the TCP performance by describing the amount of required window. In the case, we have low BDP. Since we have no delay, small round trip time (RTT). Therefore, the buffer is very large.

Analysis 1.2

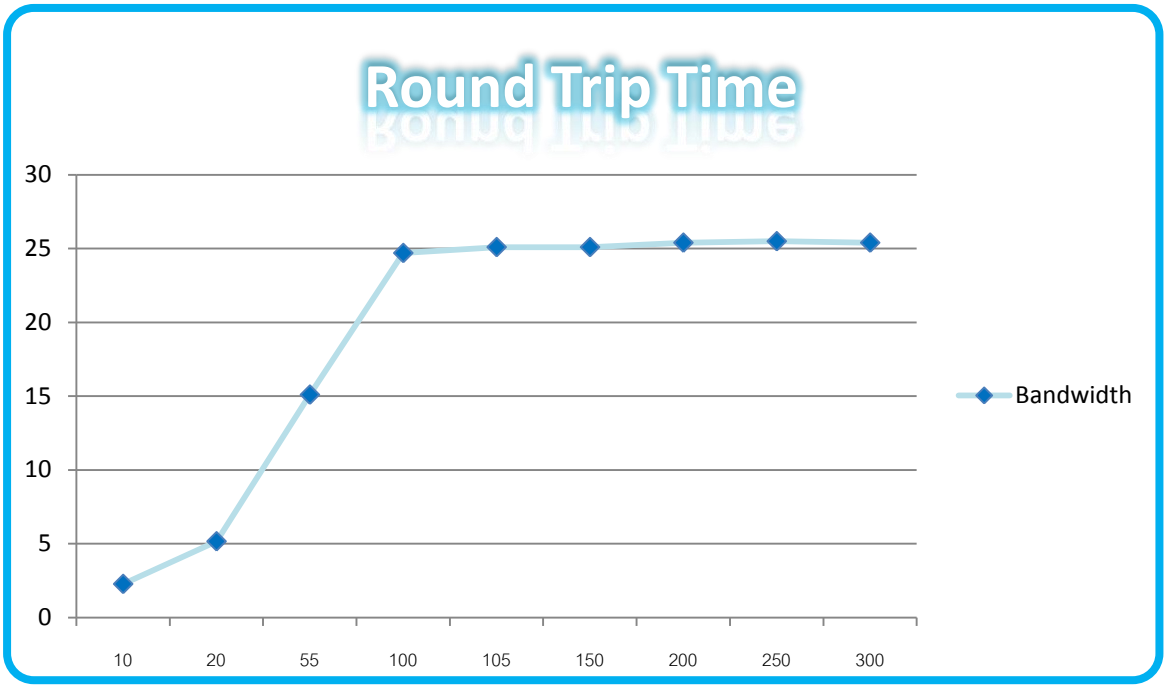


Figure 3

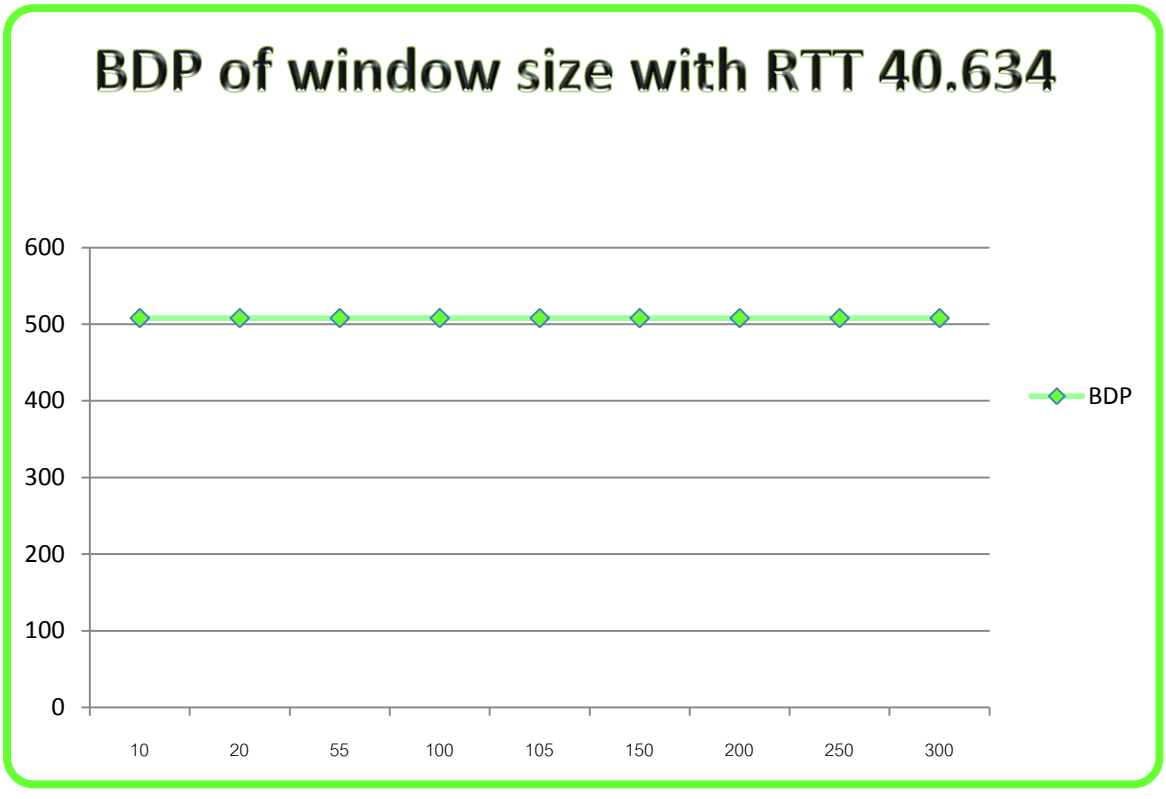


Figure 4

Since the delay has been setting before doing anything, the result is visualized that the bandwidth has been reduced at the same receive window size. For example, when the receiver advertises the receive window approximately 55 KB, the Bandwidth of the Figure3 is 93.6. Whilst the bandwidth of the Figure2 is merely 15.1, that is, since the table2 in appendix, we have set the RTT or the delay which data transverse both forward and backward.

To get full TCP performance the TCP window needs to be large enough to accommodate the Bandwidth Delay Product which is the product of bottleneck link bandwidth and the round trip time by using the formula as following:

BDP = data rate *RTT

RTT can be obtained by using ping command.

In this experience, we have used the 1 MB of link data rate. Moreover, we have set up the delay to be 40 ms

RTT that we obtain from ping command is: 40.634 ms

BDP = $100x10^{6}X\ 40.634X10^{-3}$

= 507.925 KB

For TCP, the BDP describes the amount of buffering is required in the sending and receiving host. By default, the largest buffer is 64 KB. If the BDP is small, that default value can be used. For the large BDP, it requires larger buffer.

According to this scenario, by default, the largest buffer is 64 KB. According to Figure4, we can explicitly find that BDP is approximately 507.925. Therefore, the buffer is limited.

To get full TCP performance the TCP window needs to be large enough to accommodate the Bandwidth Delay Product which is the product of bottleneck link bandwidth and the round trip time by using the formula as following:

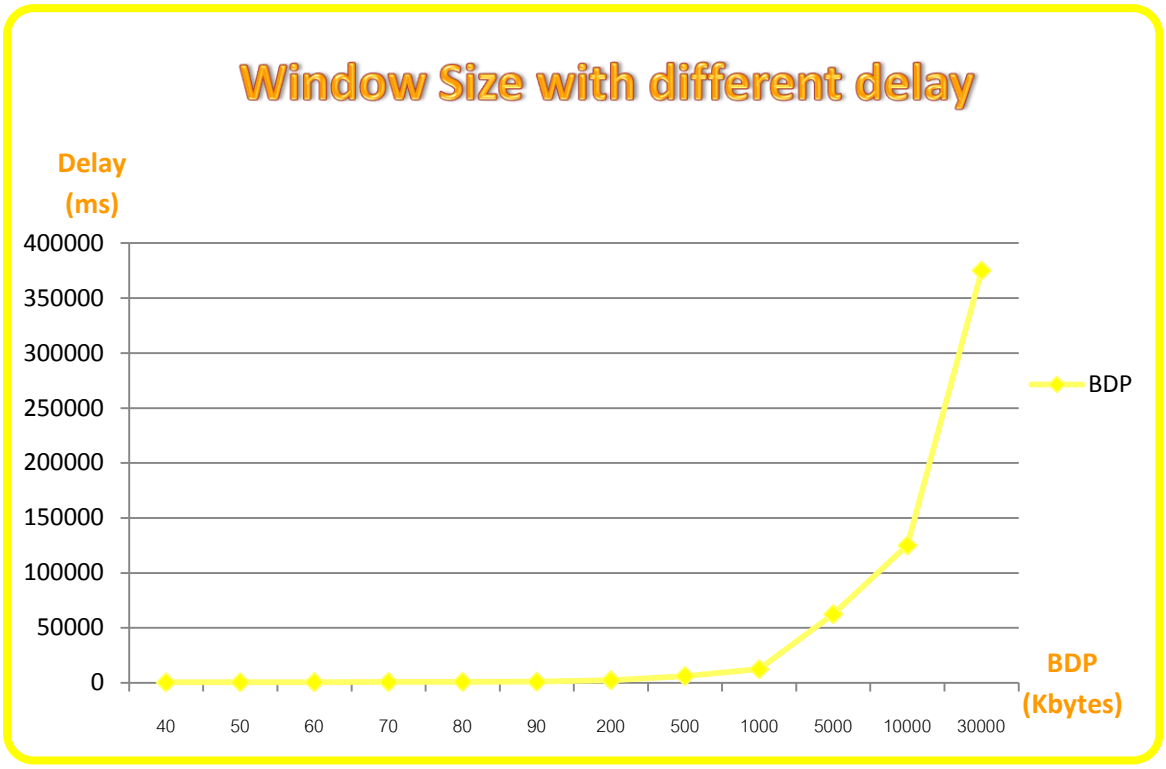
BDP = data rate *RTT

RTT can be obtained by using ping command.

The BDP result can be illustrated in table2

It can be explicitly seen that increasing the delay, can produce the increasing Round trip time. That is, the Bandwidth delay product is also increase.

Window Size 125 Kbytes with RTT 80.586 ms



Analysis 1.3

According to the table 3, we have set the window size to be constant. (Default value is 115 KB), and try to change the delay, measure the BDP value and see the result from the experiment.

We can explicitly find that the different of delay can make the TCP throughput is difference. For small delay, the throughput that we get is greater than the larger delay. According to the round trip is the length of time it takes for a signal to be sent plus the length of time it takes for an acknowledgment of that signal to be received. It measures current delay on the network. Modification of delay in this situation can make the round trip time changes. Consequence, it can lead to the varying of Bandwidth delay product (BDP).

Change in length

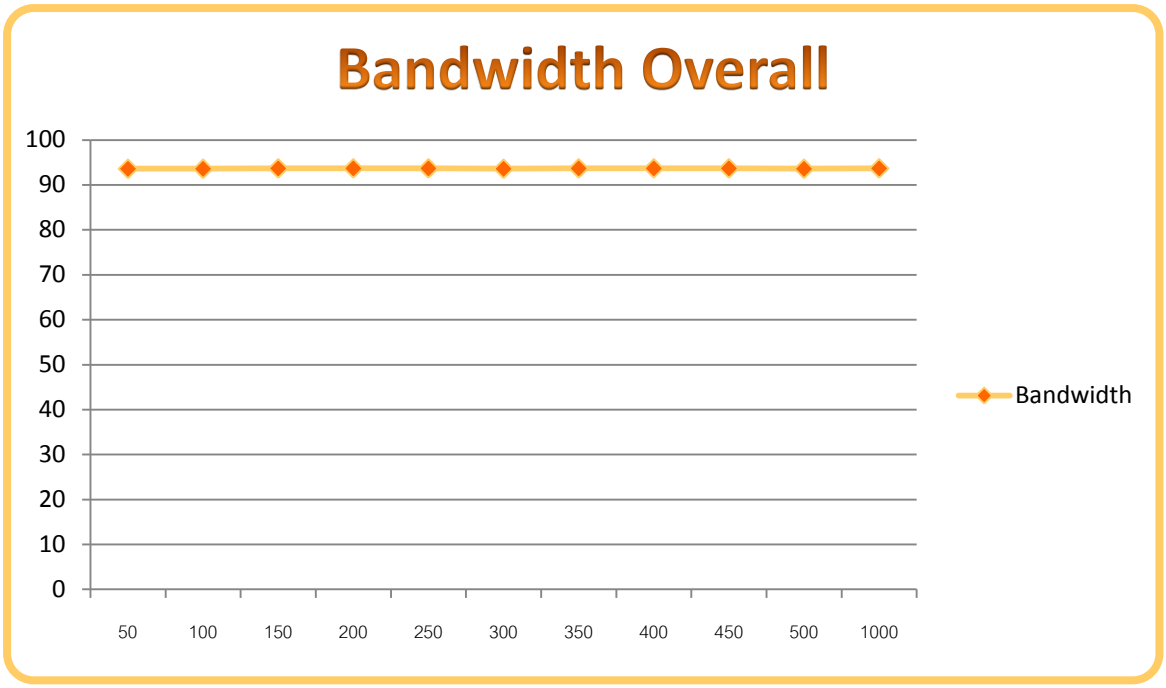


Figure 3:

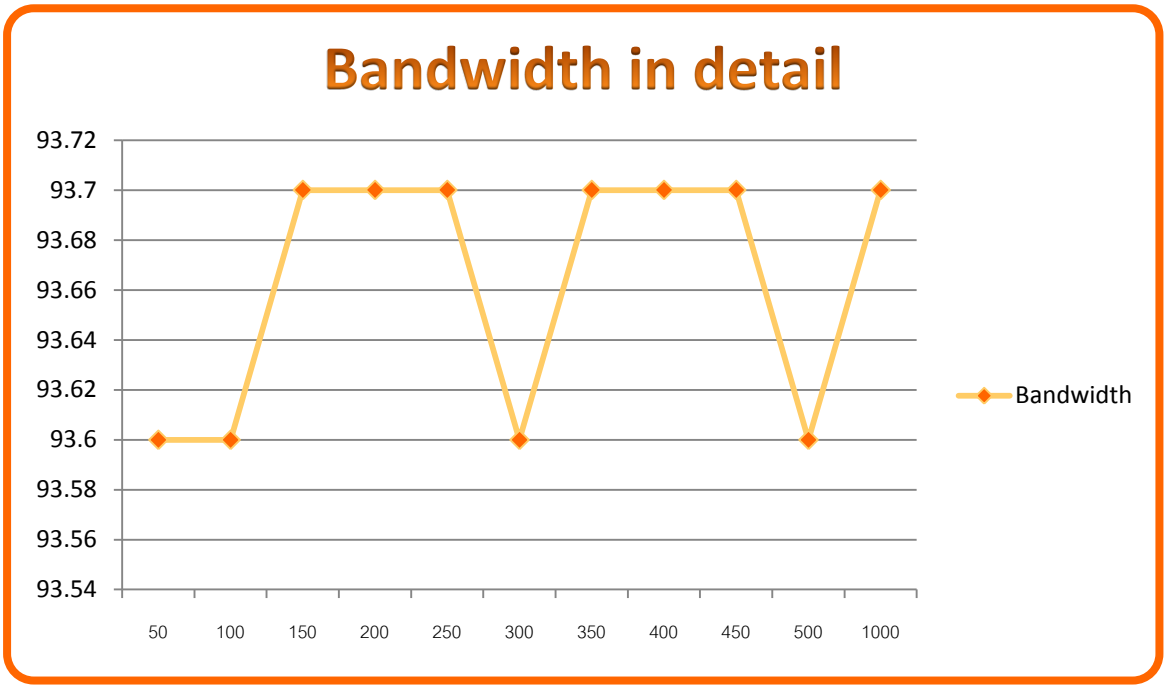


Figure 4:

For the second task, we have measured to see how much memory is allocated to send and receive traffic flow buffers (buffer length) or we can say that it is the size of the data which is limited to the application. As we can see from Table2, with constant window size, for this scenario, we try to change the value of length which is the amount of data each time instead. In term of congestion scenario, it can be classified in the case which a router has an infinite buffer. The link that we use has 100 MB for link capacity.

According to Figure3, we will see that even we try to increase the number of length ,the throughput or the output capacity are limited approximately 93.7 Mbits., which mean that a router has finite buffers.

According to this scenario, we will see that if the buffer is full and new packet arrives, those packets will be dropped which lead to retransmissions.

To conclude, the length or the buffer size in this scenario may not affect the TCP performance since in this case the buffer size is limited at 93.7 Mbits/s. Even we try to increase the length, however, it still does not affect to the TCP performance.

2.) Single TCP session; varying network/link conditions.

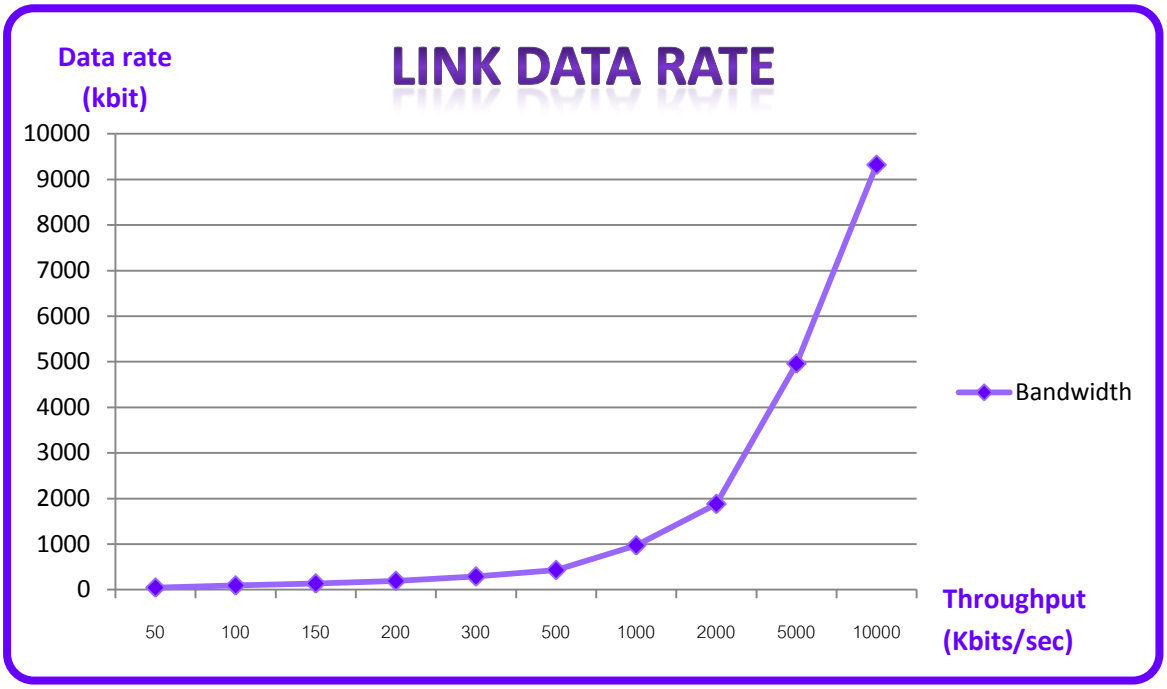


Figure5

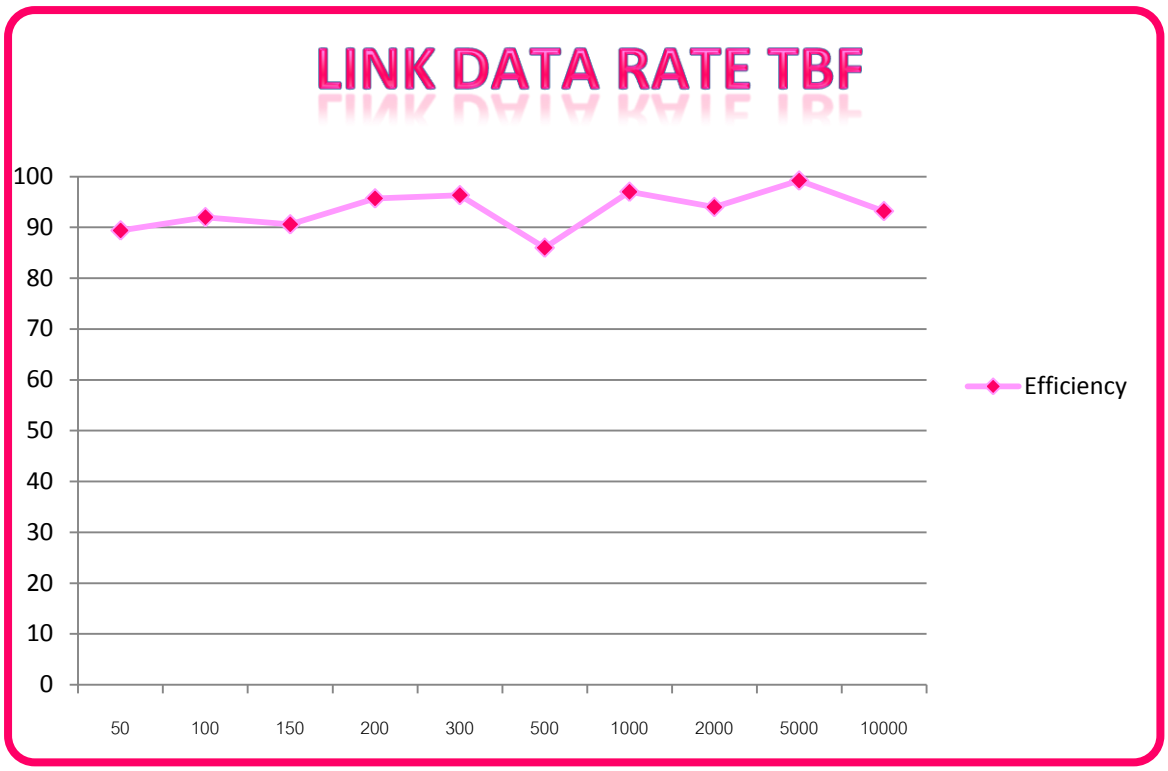


Figure 6

Analysis: link data rate

According to TBF description, all packets that fit into the buffer are immediately sent. However, if the buffer is full, incoming packets are queued and wait for their queue. By definition, TBF is the maximum data rate or the capacity of the network.

Idealistic, the TBF should be equally to the throughput. However, we don't expect to be get 100% efficiency. The number of throughput which closes to the 100% may be more expectation.

From the table , even lower TBF, we can get high efficiency. High efficiency does not need to be high TBF. In the meanwhile, even it doesn't reach to the maximum bandwidth, it still be efficiency. Moreover, for lower TBF, it can get high efficiency. Therefore, we don't expect to get high data rate, but only expect to get high efficiency as much as we can.

According to the Figure 5, we can seethat the trend is unstable. The graph is oscillatory changed.

Moreover, For the Figure 6, we can see that TBF change, TCP scales are also change. The throughput is likely to increase when we the throughput increase.

To conclude, We don't expect 100% efficiency or no overhead. However, we only expect high efficiency as much as we can. Or we can say that we only expect just small overhead. Lastly, even high TBF but less efficiency, which means that it is meaningless to get high TBF.

Packet drops

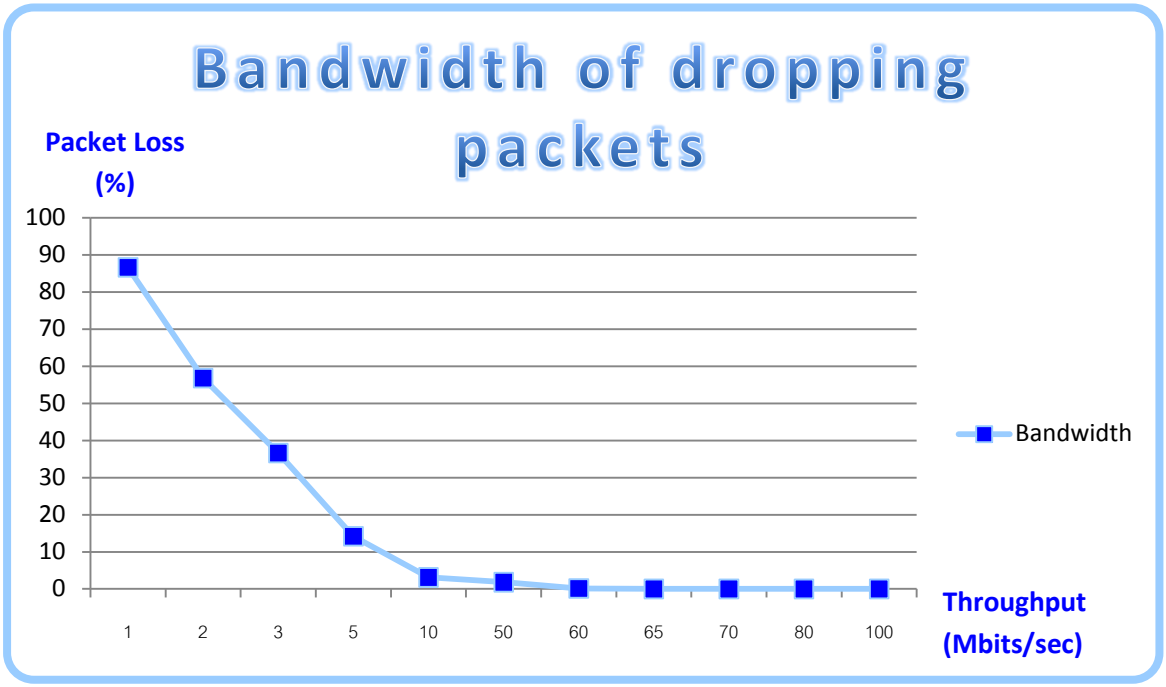


Figure7

Analysis: packet loss

Theoretically, packet loss is the failure of one or more transmits packets to arrive at their destination. According to network congestion, TCP sender will assume a loss may cause from either TCP sender timeout or TCP sender receives 3 duplicate ACKs. From the second task, we will see that the losses are usually due to congestion of the network and buffer is overflow. For this task, we will try to increase the loss and see the affect from packet loss.

According to Figure7, we try to increase the percentage of packet loss; the output shows that the more we increase the percentage of packet loss, the less data transfer and throughput that we will get.

Since the TCP has the flow control feature which ensure the packet that we send will not overflow the receiver. If no congestion detected, the TCP sender will increase the sending rate or doing additive increase. However, in this case, the packet loss is detected; therefore the Multiplicative Decrease will be used. With this algorithm, TCP sender decreases its sending rate which can be illustrated by Figure 7.

The type of loss event for this scenario can be assumed to be the loss which detected by a timeout which is more congestion than loss which detected by three duplicate ACKs. Since, we can find that when we increase the percentage loss reaching to 65%. The host TCP sender will not receive ACK from TCP receiver.

However, sometime that loss may cause from the hardware problem or due to the link error.

To conclude, according to the host sends data too much which lead to buffer overflow. The buffer overflow can enhance the packet drop. Moreover, the more packets drop the more retransmission and the reduction of TCP throughput. If too much packet drop, it can lead timeout event occurs which happen in this situation.

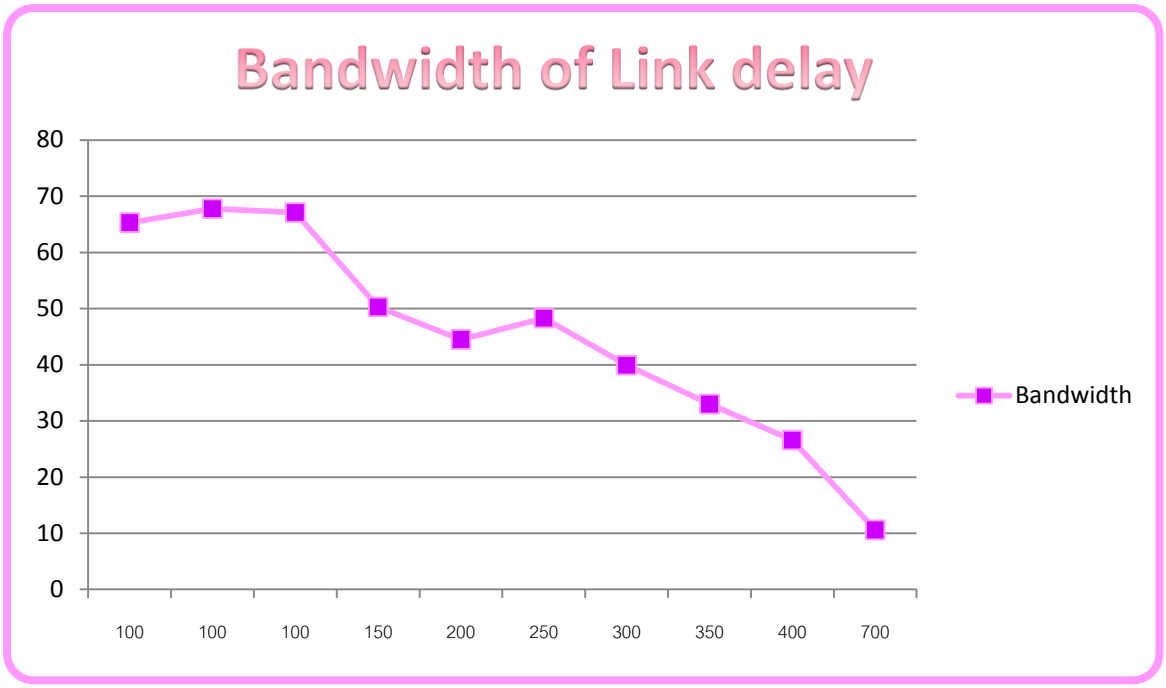


Figure 8

In real world, link delay can be dominant for throughput determination. The larger delay which can be demonstrated in Figure 8 ,can produce the lower throughput. Not only throughput that will be diminished, but also the data transfer is also be diminished. Since, the throughput or effective bandwidth is proportional to the data transfer. In case of large queuing delay, sender must perform retransmission in order to compensate for drop packet/ loss packet.

3.) Multiple TCP sessions.

(2 TCP clients)

Figure 9

3 TCP clients

Figure 10

4 TCP Clients

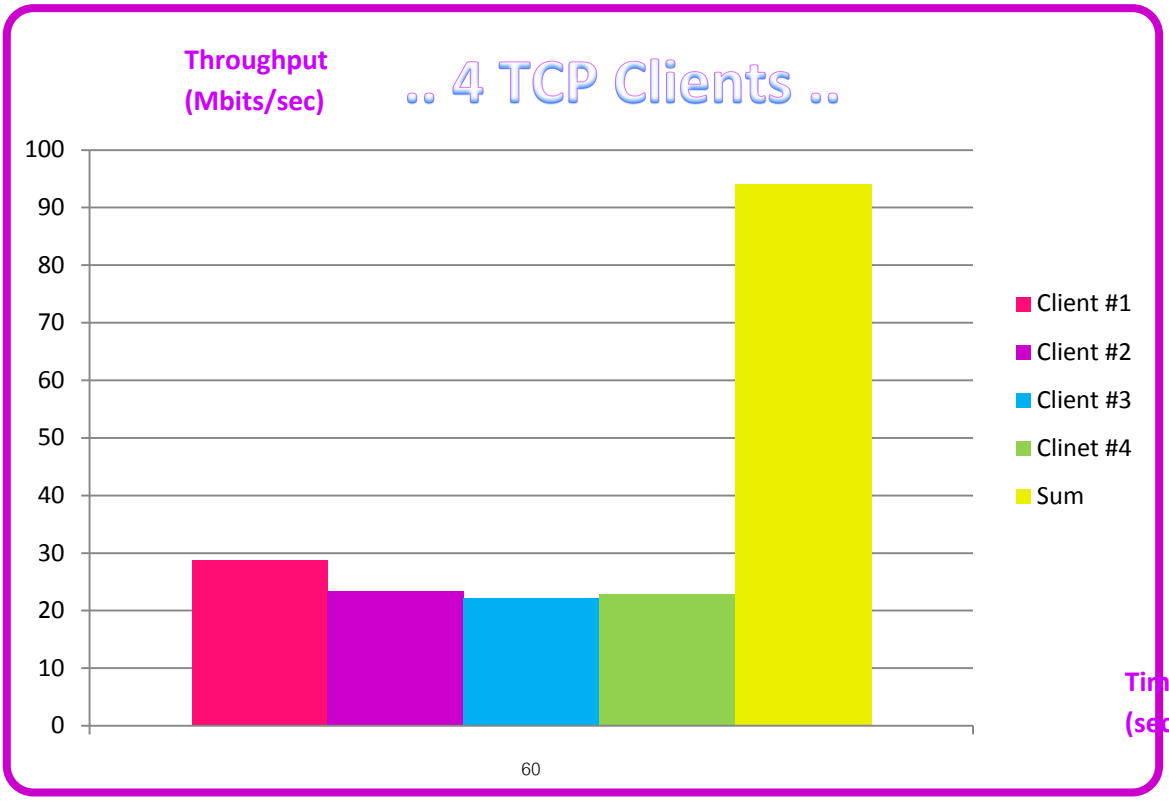


Figure 11

Theoretically, for single TCP session, the throughput or effective bandwidth will not be shared. The client can get the higher throughput than shared bandwidth. However, the throughput will not be dramatically changed too much at the level of maximum throughput. However, as we can see from this scenario, that the throughput is shared by multiple TCP session. The more TCP sessions, the fewer throughputs the client will obtain.

With multiple TCP sharing the same bottleneck link do not need to get the same bandwidth or throughput. As we can see from Table 5, the first client gets higher throughput than the second client. Similarly, the throughput of Figure 9 , 10,11 are unequally shared.

With time is constant, two clients are not get equally shared throughput. This can be illustrated by Figure 7.

1 TCP clients and 1 UDP client

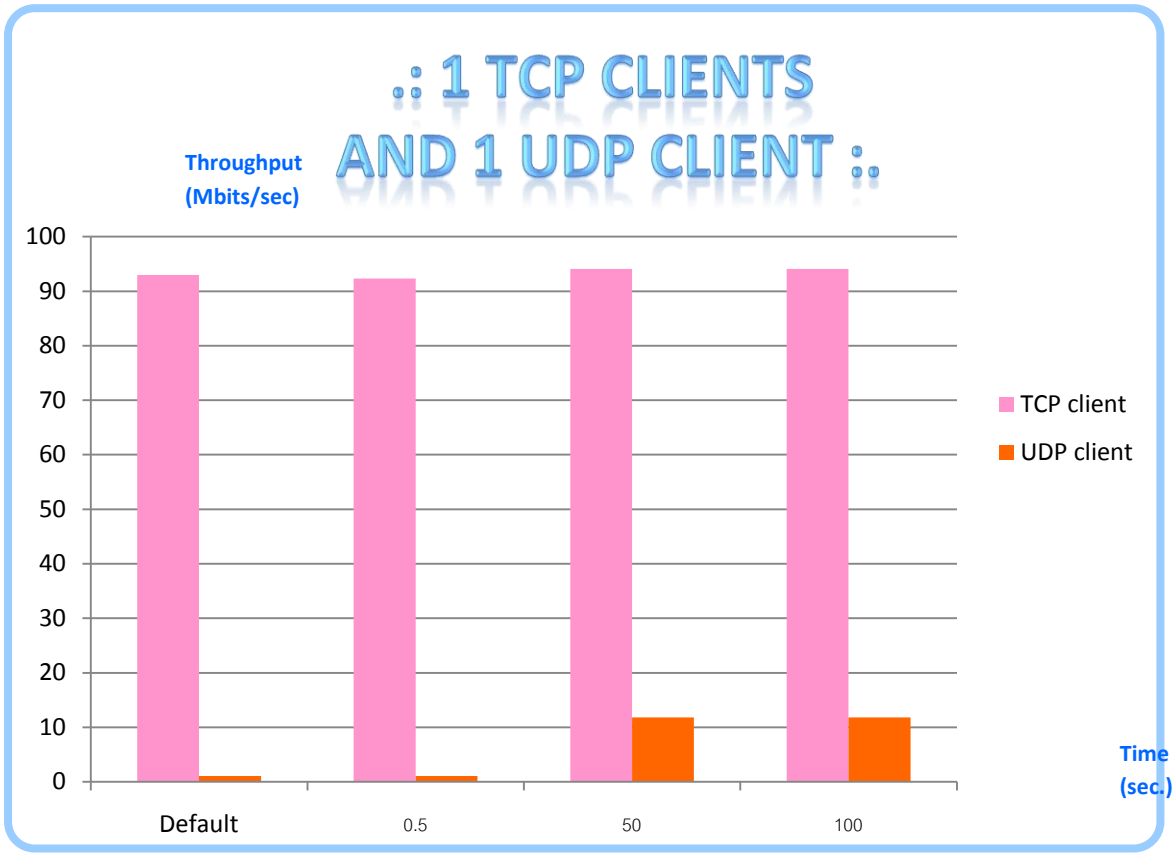


Figure 12

2 TCP clients and 1 UDP client

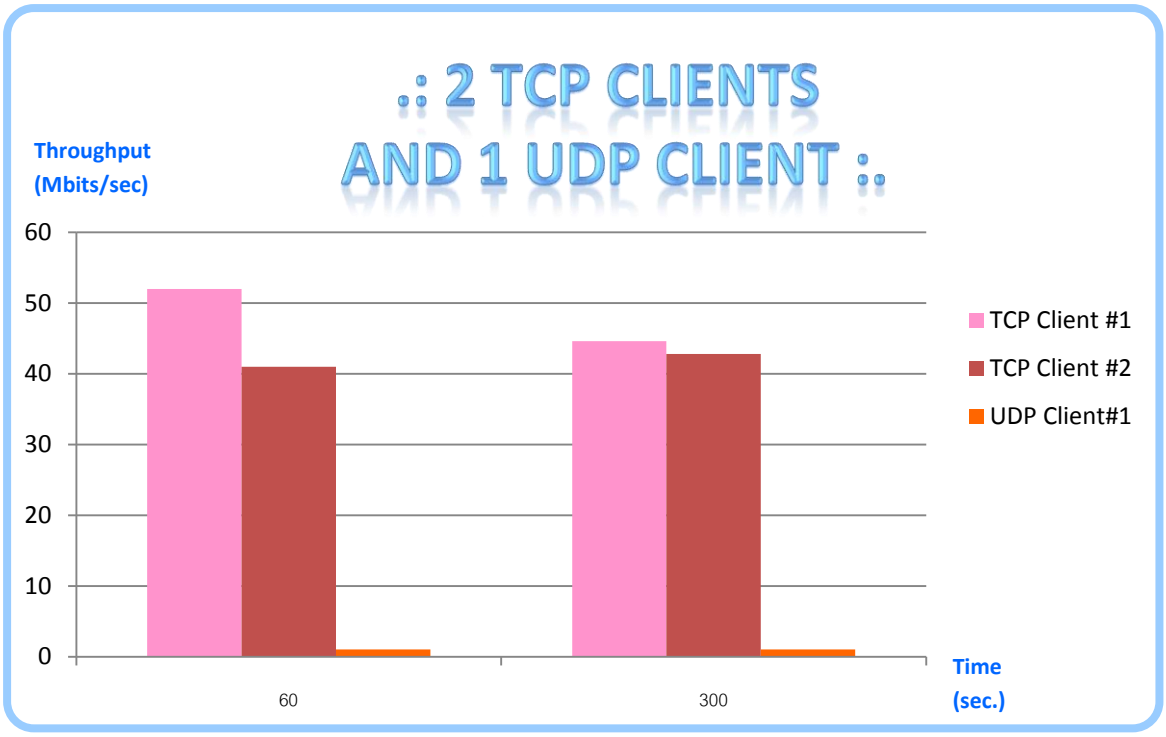


Figure 13

3 TCP clients and 1 UDP client

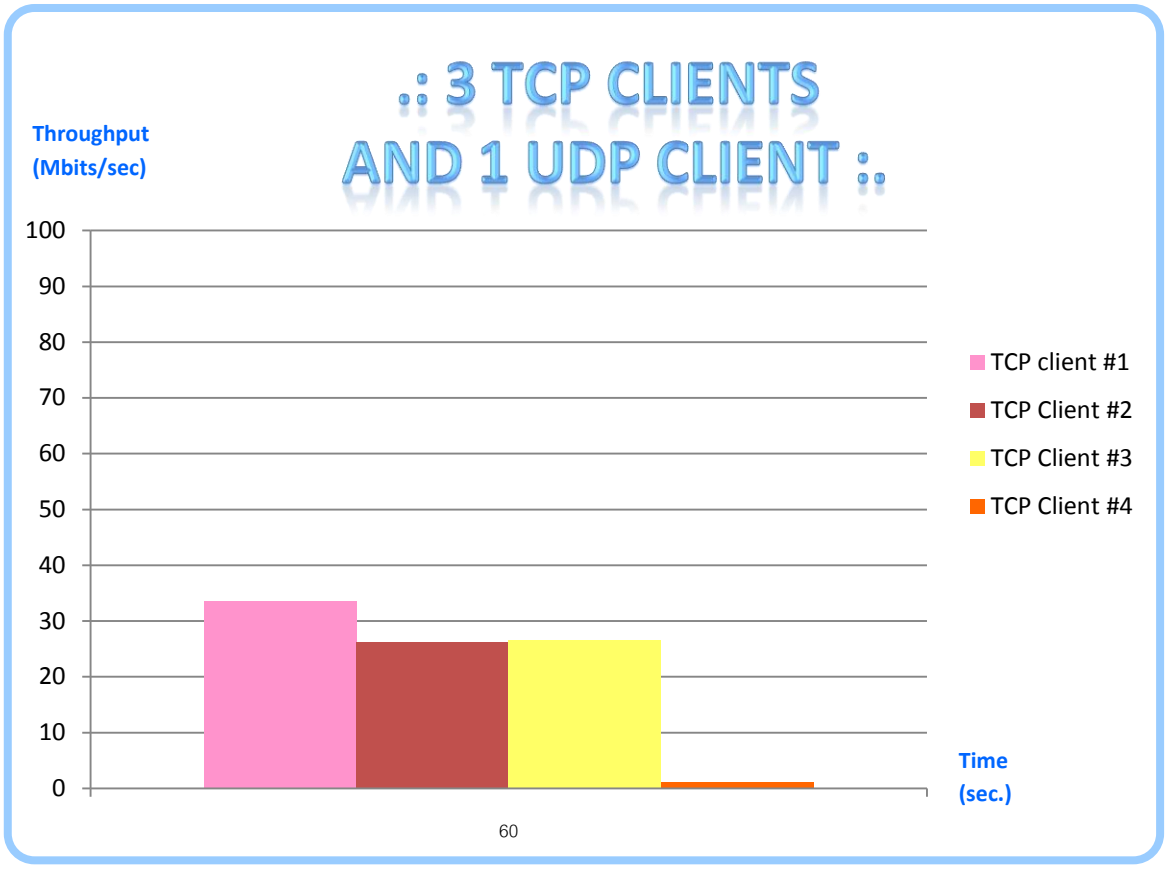


Figure 14

TCP Client and 2 UDP Clients

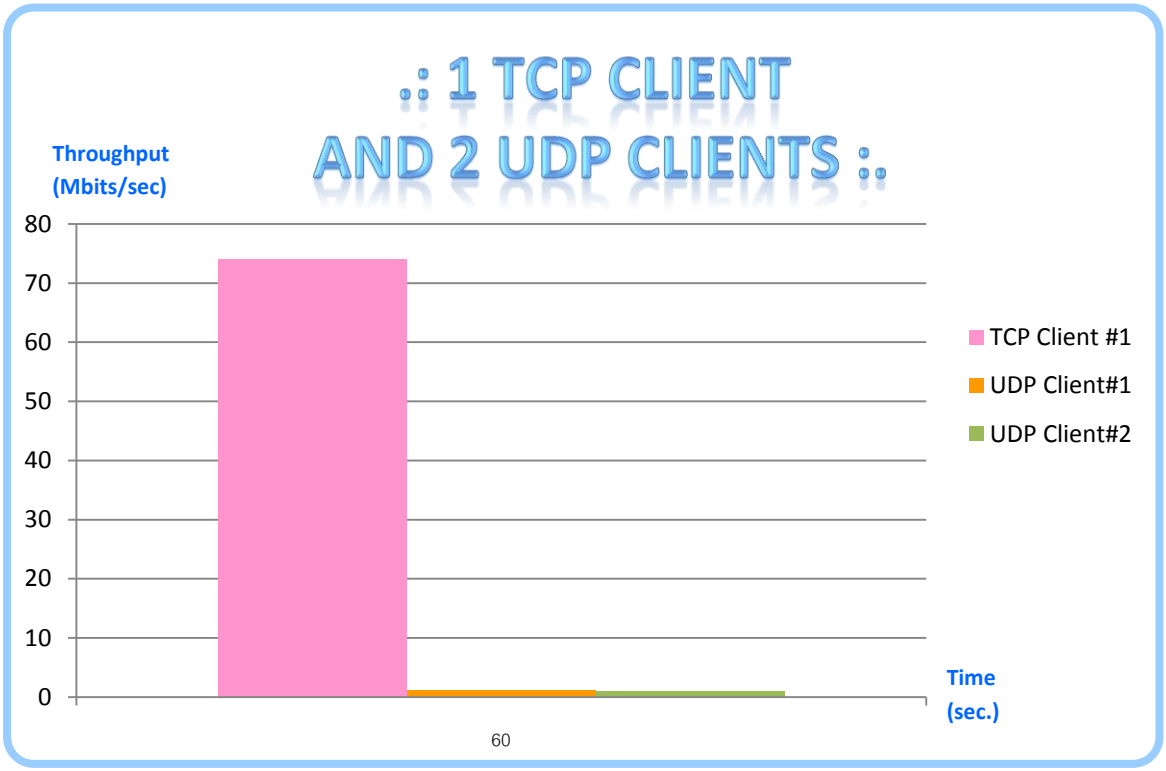


Figure 15

For table 8-11 we can see that TCP will always transfer more data than UDP because TCP will sent the

data by fragment data in many packets and will be merged those packets at the destination and TCP and will try to sent data as fast as possible. TCP guarantee that data will be received at the destination by checking at the destination that data was received or not. If there is some data has some segment has been dropped before reaching the destination, TCP will retransmit those missing segments. With this capability, therefore it will make TCP more reliable than other protocol in transport protocol. Moreover, TCP is able to transmit more data than UDP since the occurence of packet loss is less han UDP owing to having congestion control to regulate the sending rate. But for UDP it sent data on demand and it doesn't have the method to check that it will reach the destination or not, so it does not guarantee that data will be received at the destination that will make UDP get more chance of packet loss, so it will make UDP less data transfer and throughput than TCP

As you can see in table 9 and 10 if you use more than 1 TCP at the same time ,throughput will be shared to all TCPs that use at the same time by applying TCP fairness to these and if you sum throughput of all TCP it will be close to or lower from if you use 1 TCP , this can happen because some error may occur from hardware or software , for hardware it may have delay or packet loss between iperf and lan card , for software it may come from the behavior of iperf or operating system. Another reason that may affect is our output that we got is not an average since we measure only one time but the output can be variances due to many factors

But for UDP if we use more than 1 UDP at the same time such as in table 11 that using 2 UDP at the same time ,you can see that UDP will not shared its throughput because UDP doesn't has a flow control and congestion control to control when it transmit a data

Table of participation

Name	Set up	Conducting an experiment/ result	Analysis
Natchanan	33%	33%	33%
Atjima	33%	33%	33%
Hutchaganit	33%	33%	33%

`

Assignment 2

(ITS 413)

Offered to

Dr. Steven Gordon

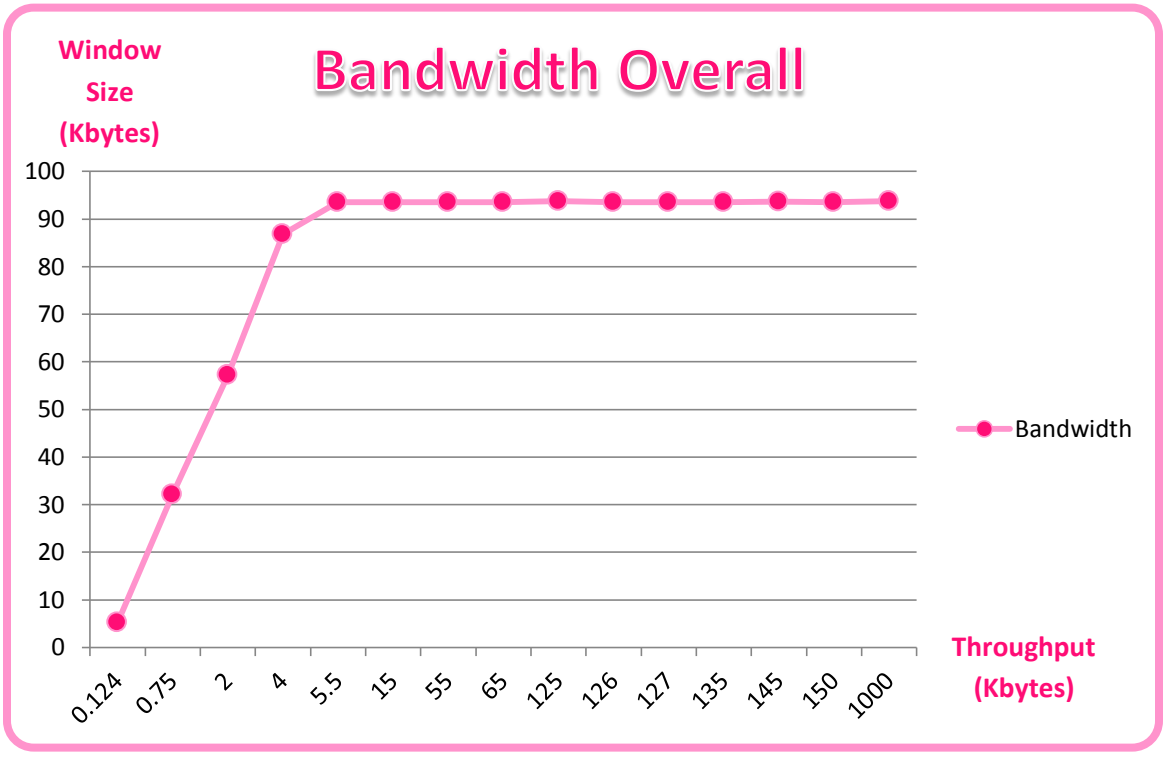
Presented By

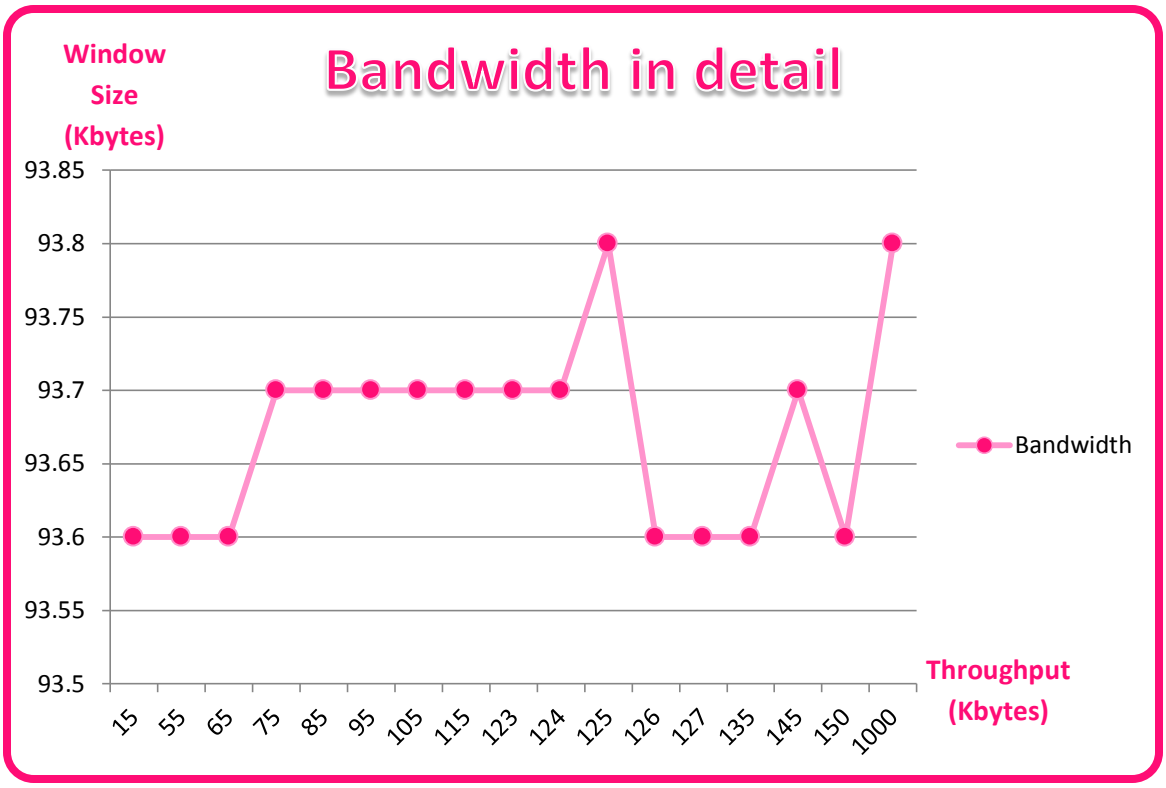
1. Mr. Natchanan Phenchan ID. 5122780885
2. Ms. Atjima Meetham ID. 5122781156
3. Ms. Hutchaganit Putiprawan ID. 5122790090

Bandwidth of single TCP session

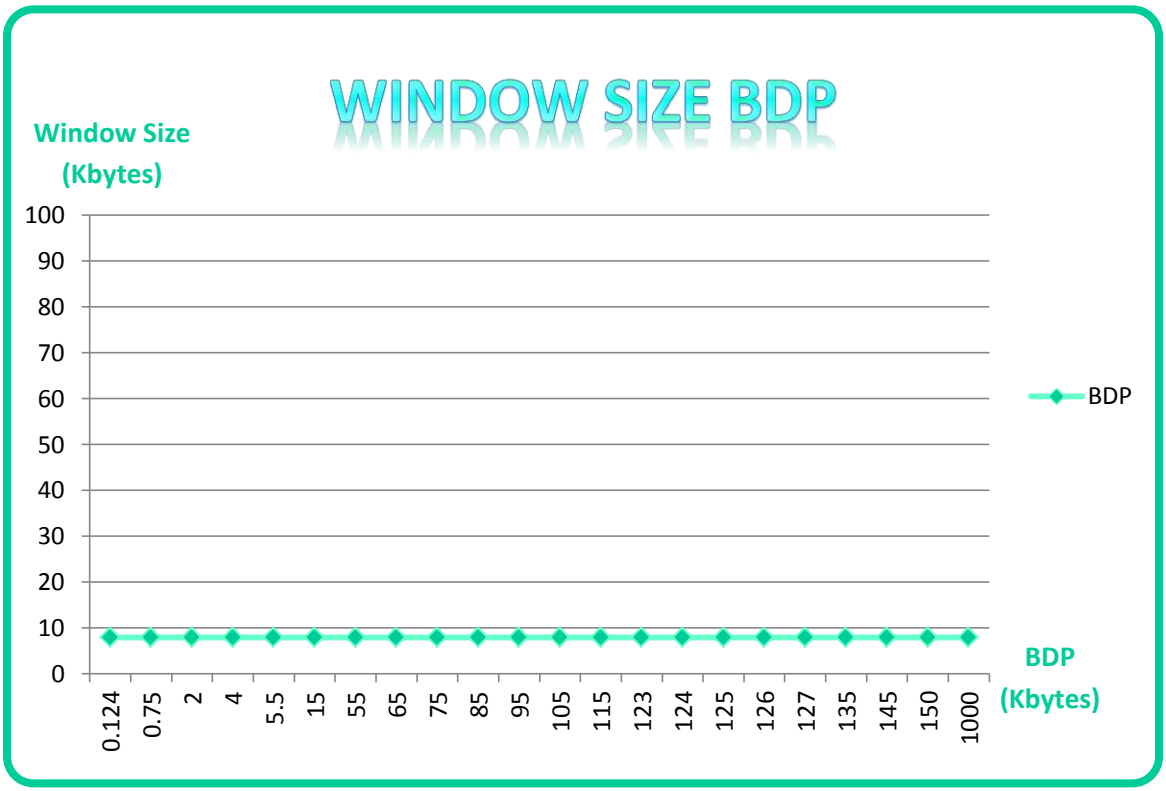
(Varying application/protocol)

- **Change in Window size with RTT 0.637 ms (in table 1)**

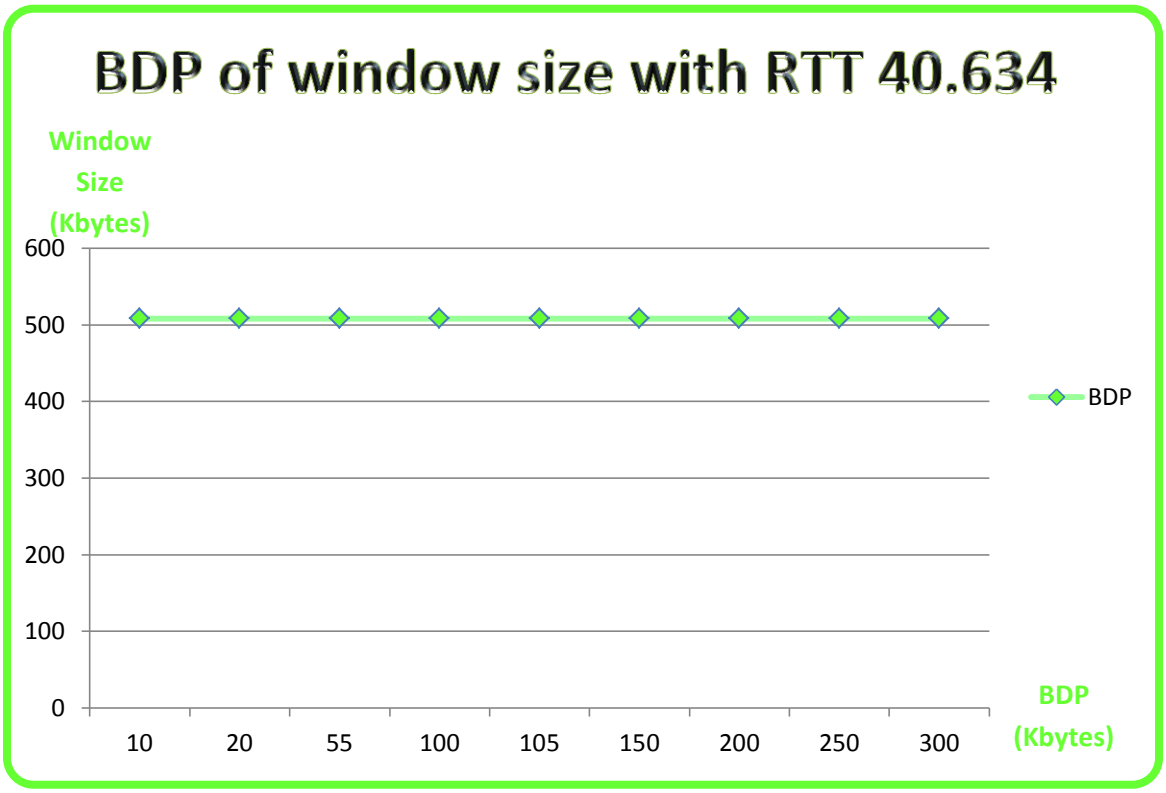



Bandwidth in detail
Window Size (Kbytes)
93.85
93.8
93.75
93.7
93.65
93.6
93.55
93.5
15
55
65
75
85
95
105
115
123
124
125
126
127
135
145
150
1000
Throughput (Kbytes)
Bandwidth

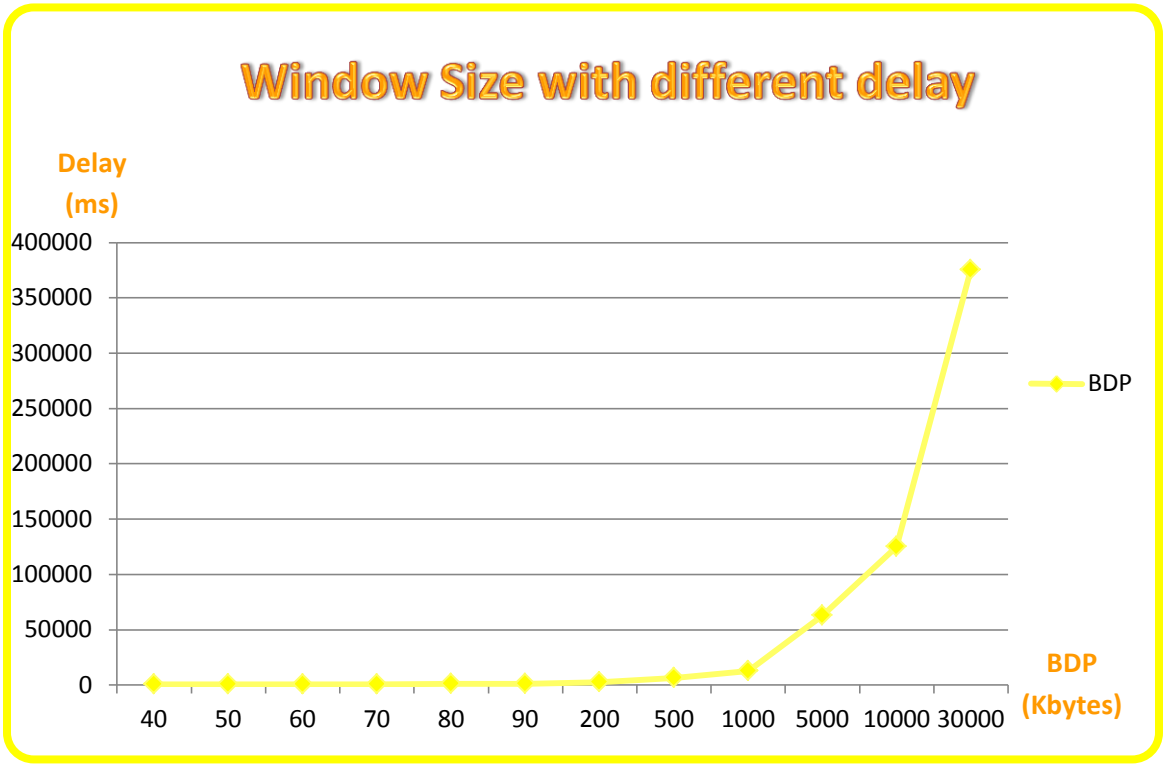
BDP of Window Size with RTT 0.637 ms (in table 1)



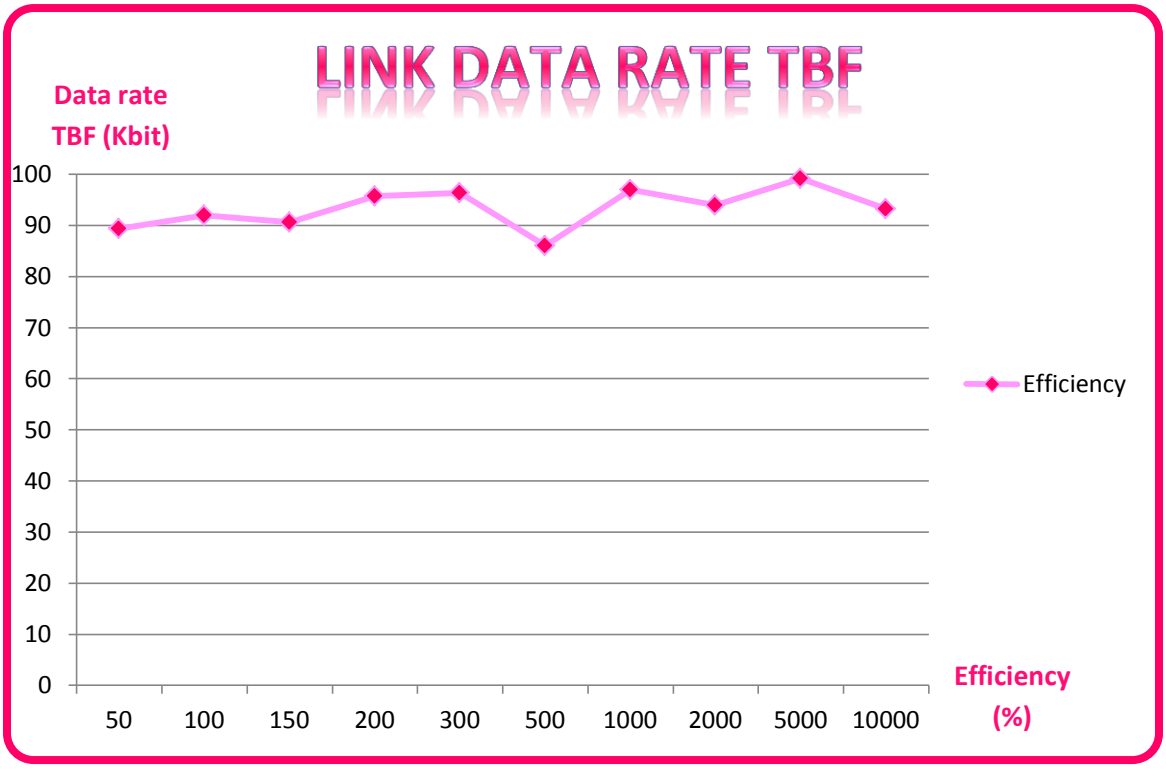
BDP of window size with RTT 40.634 (in table 2)



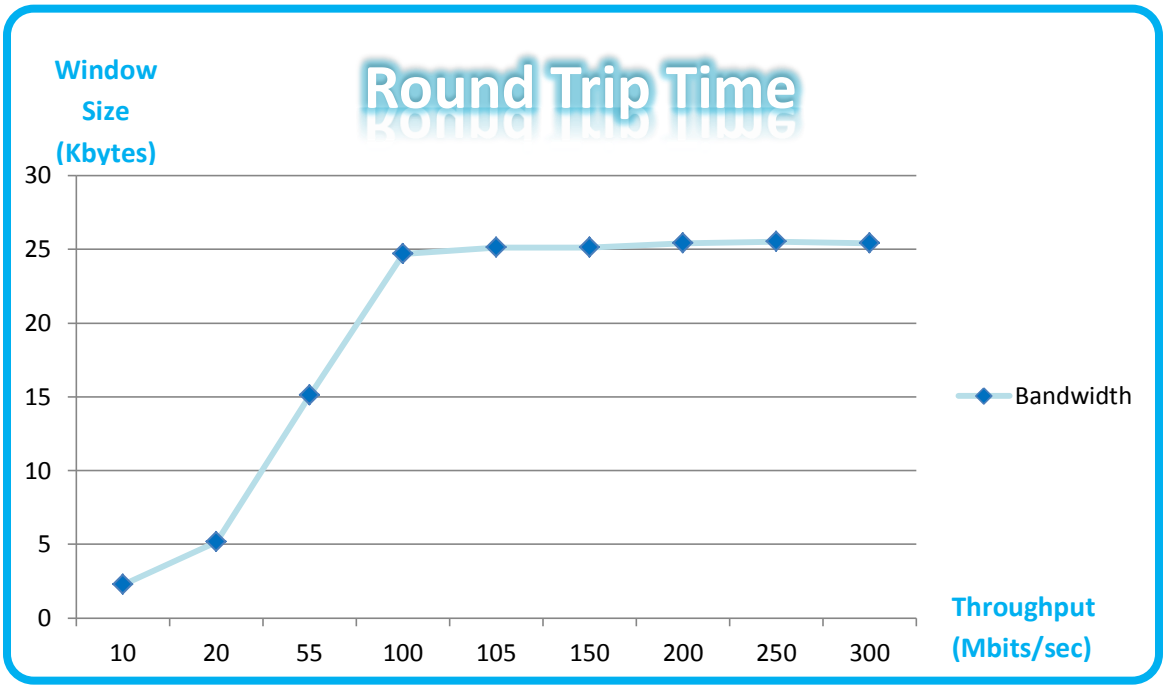
Window Size 125 Kbytes with RTT 80.586 ms (in table 3)



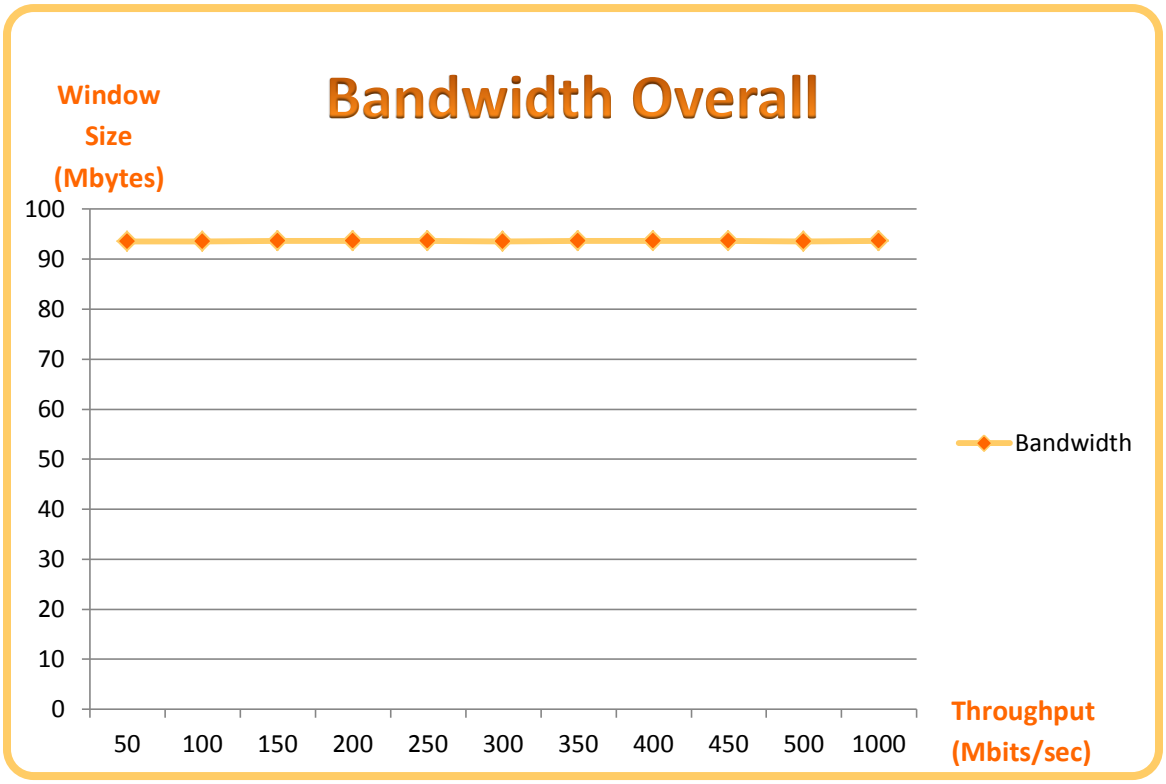
Efficiency of Link data rate TBF (in table 5)

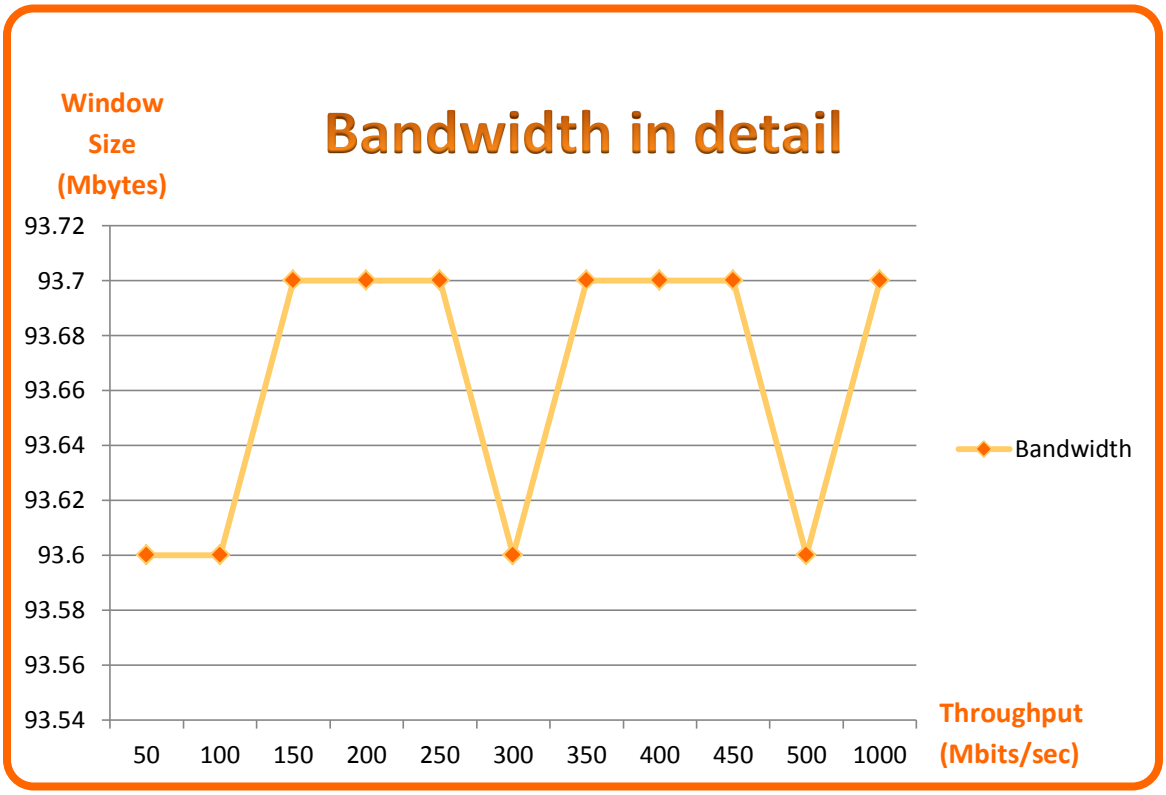


Bandwidth of window size with RTT 40.634 (in table 2)



Change in length (in table 4)

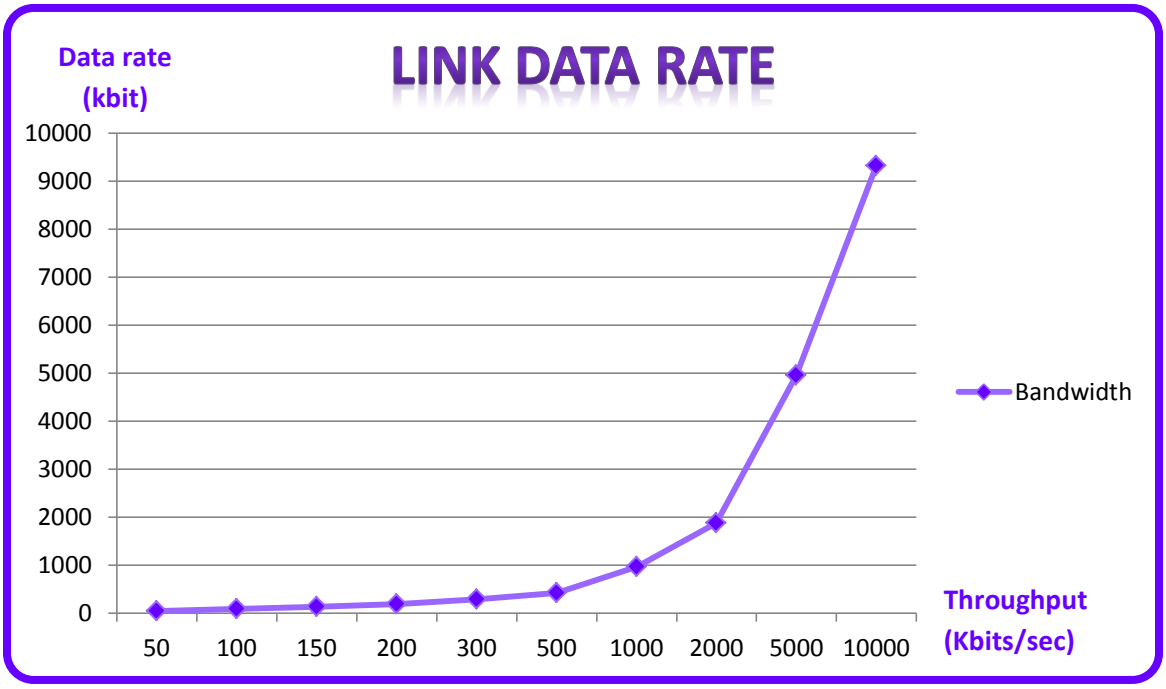




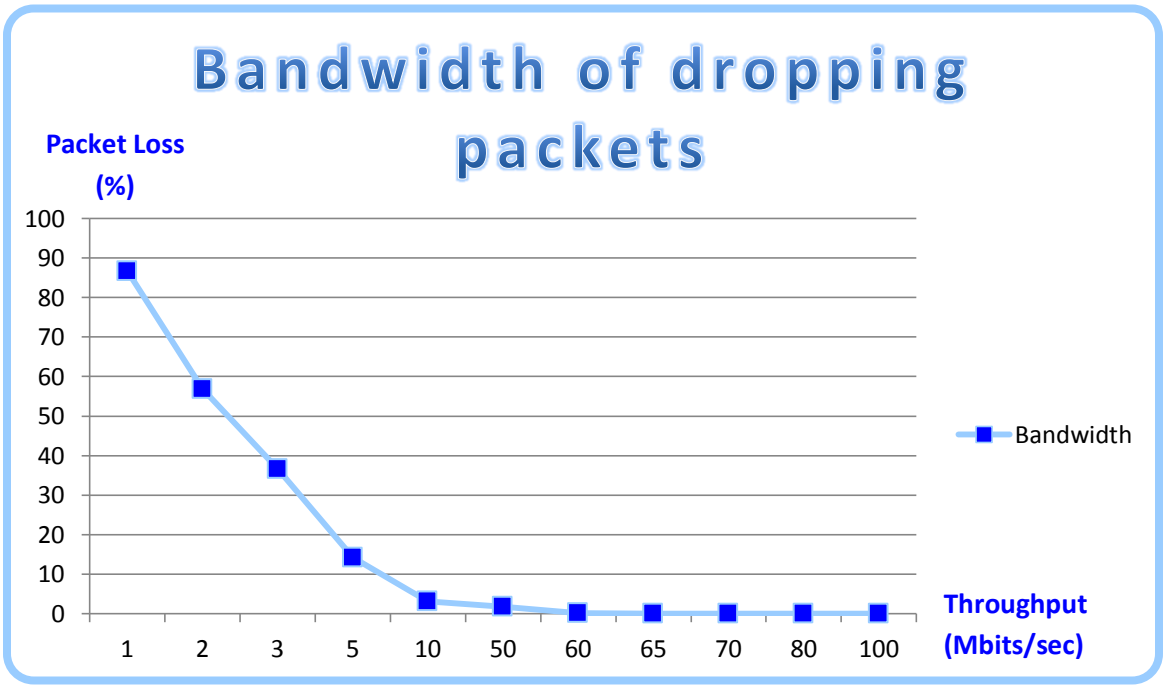
Bandwidth of Single TCP session

(Varying network/link)

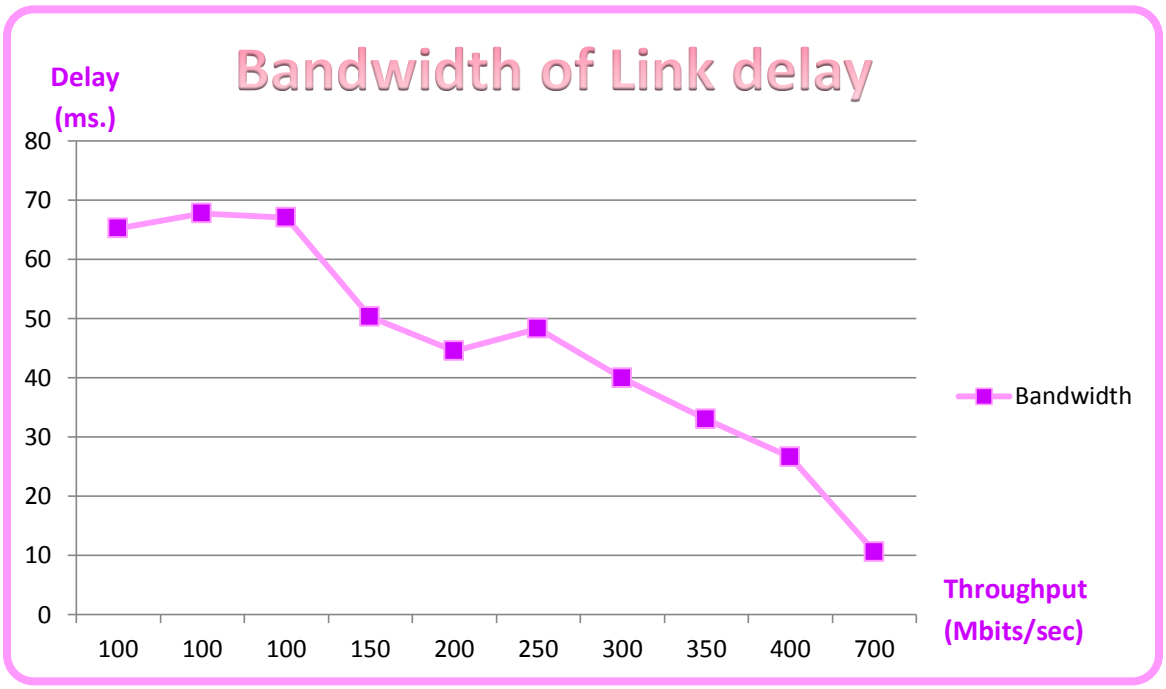
- Link data rate (in table 5)



- Packet drops (in table 6)

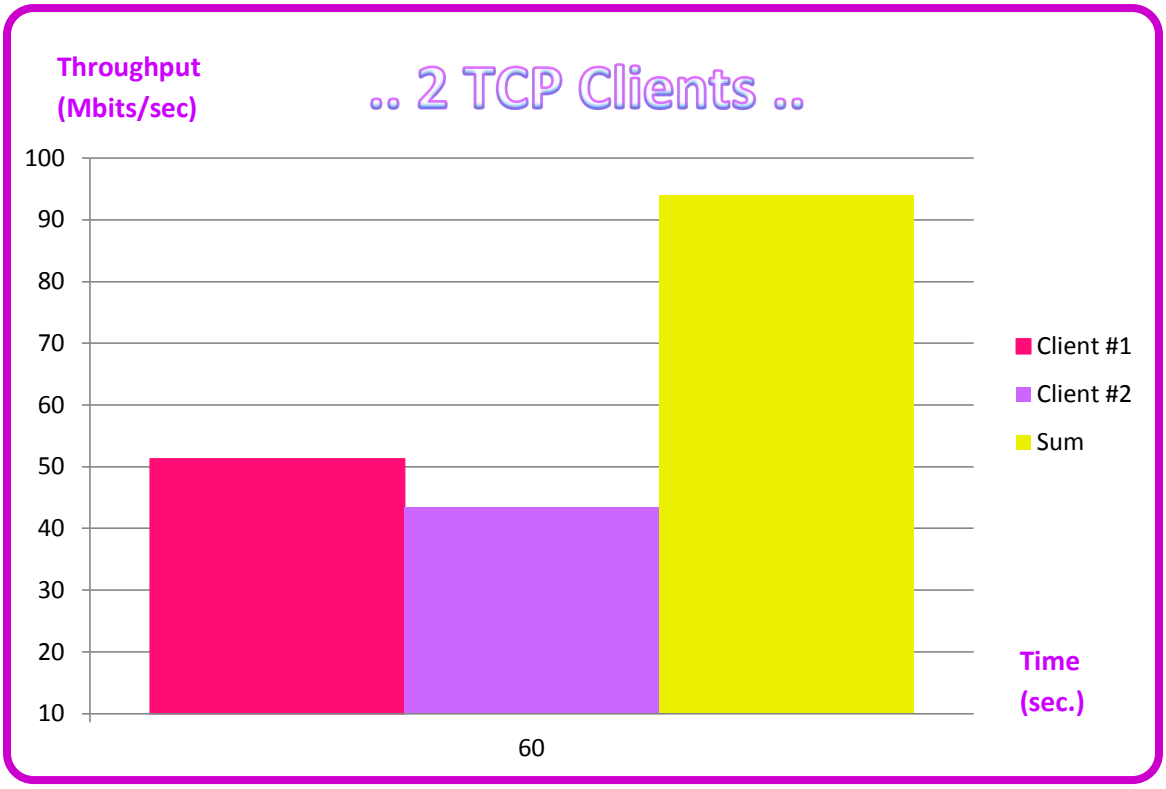


Link delay (in table 7)

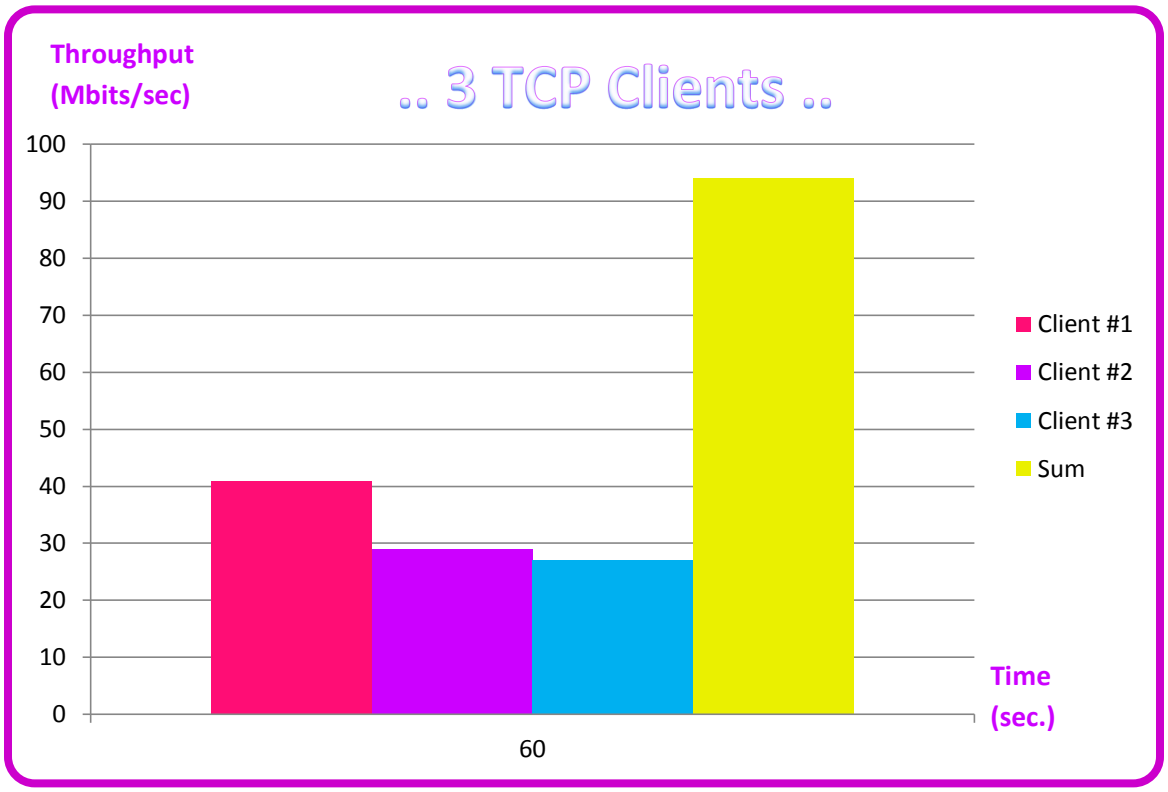


Multiple TCP sessions

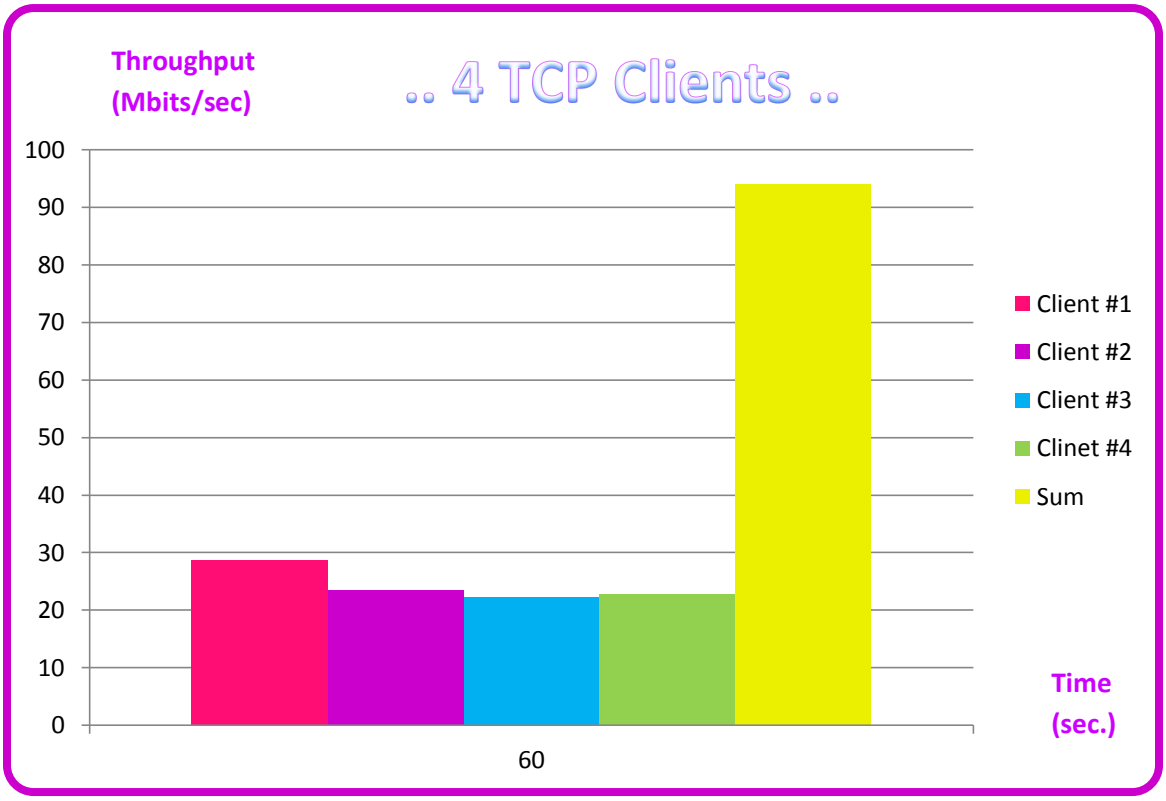
2 TCP clients (in table 8)



3 TCP clients (in table 9)

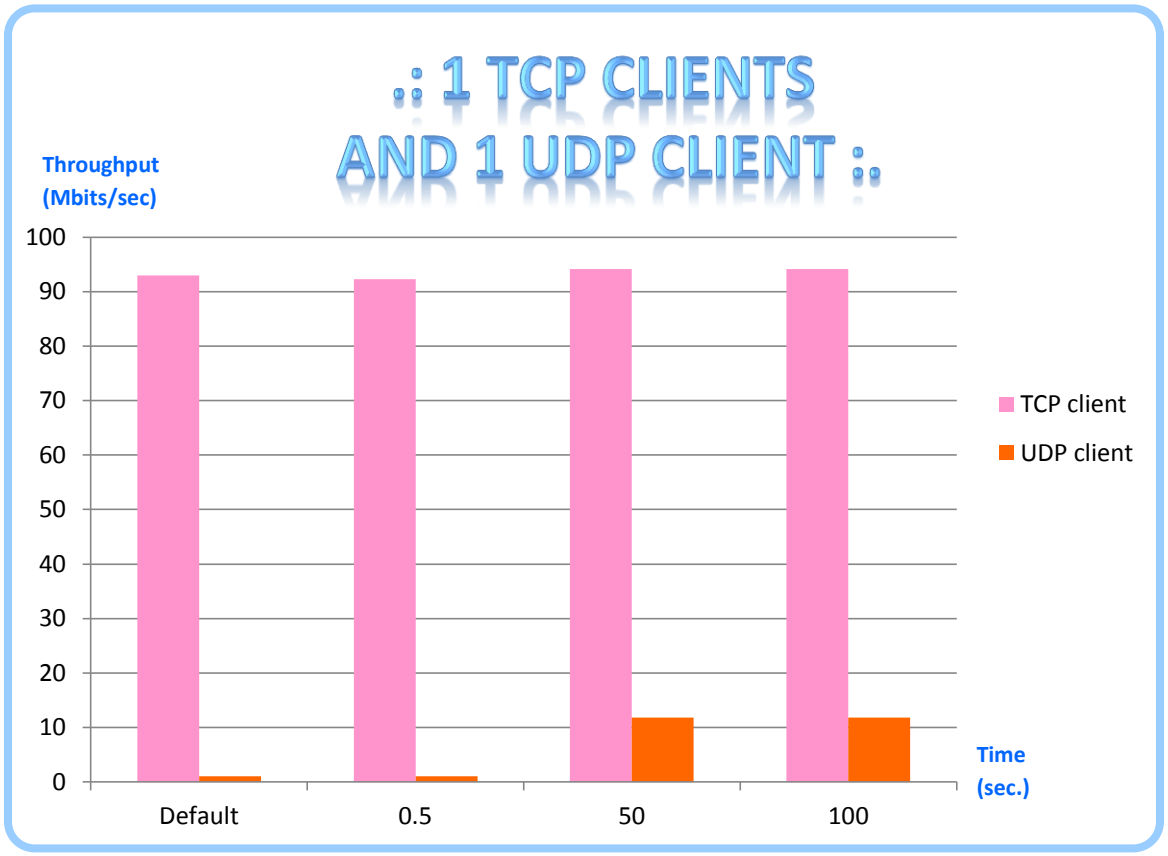


4 TCP Clients (in table 10)

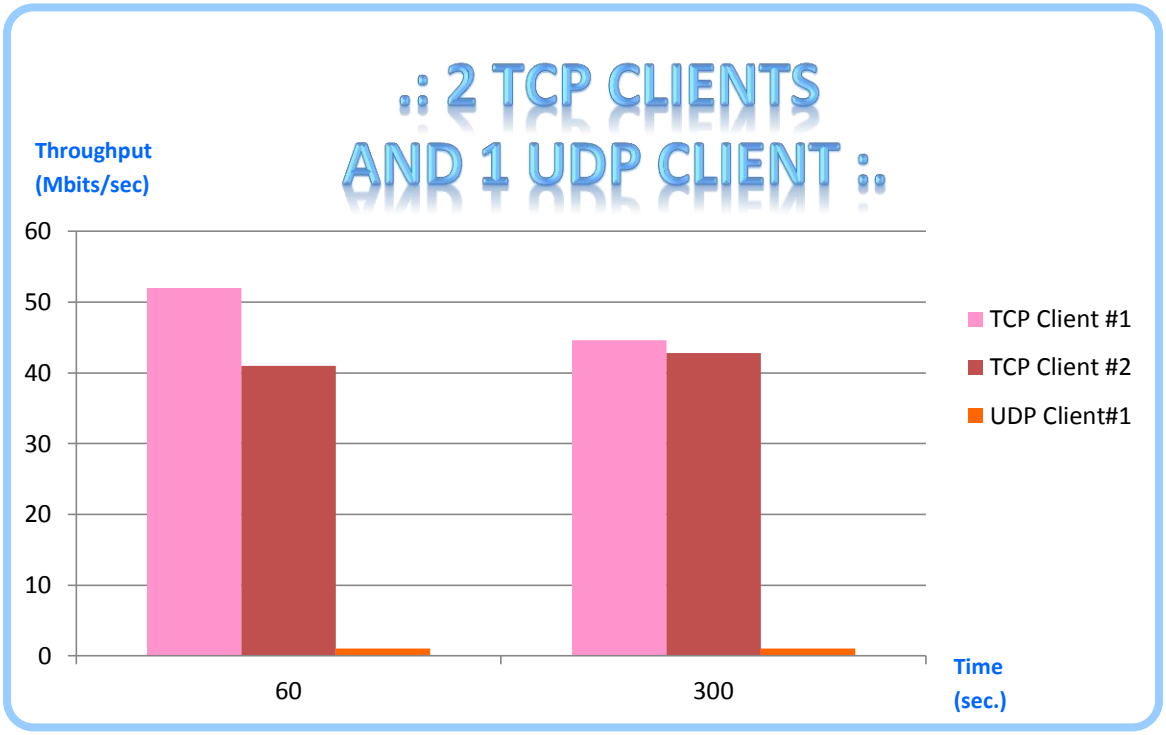


Multiple TCP sessions in presence of UDP

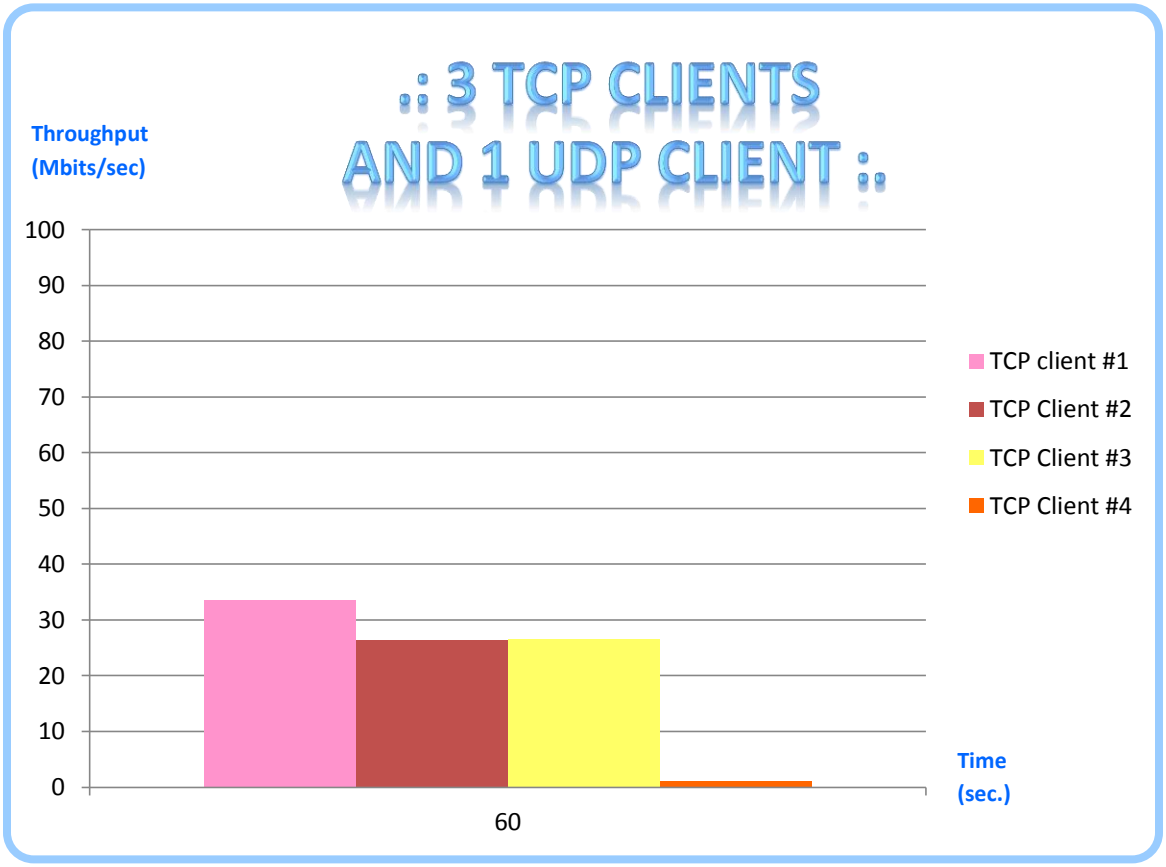
1 TCP clients and 1 UDP client (in table 11)



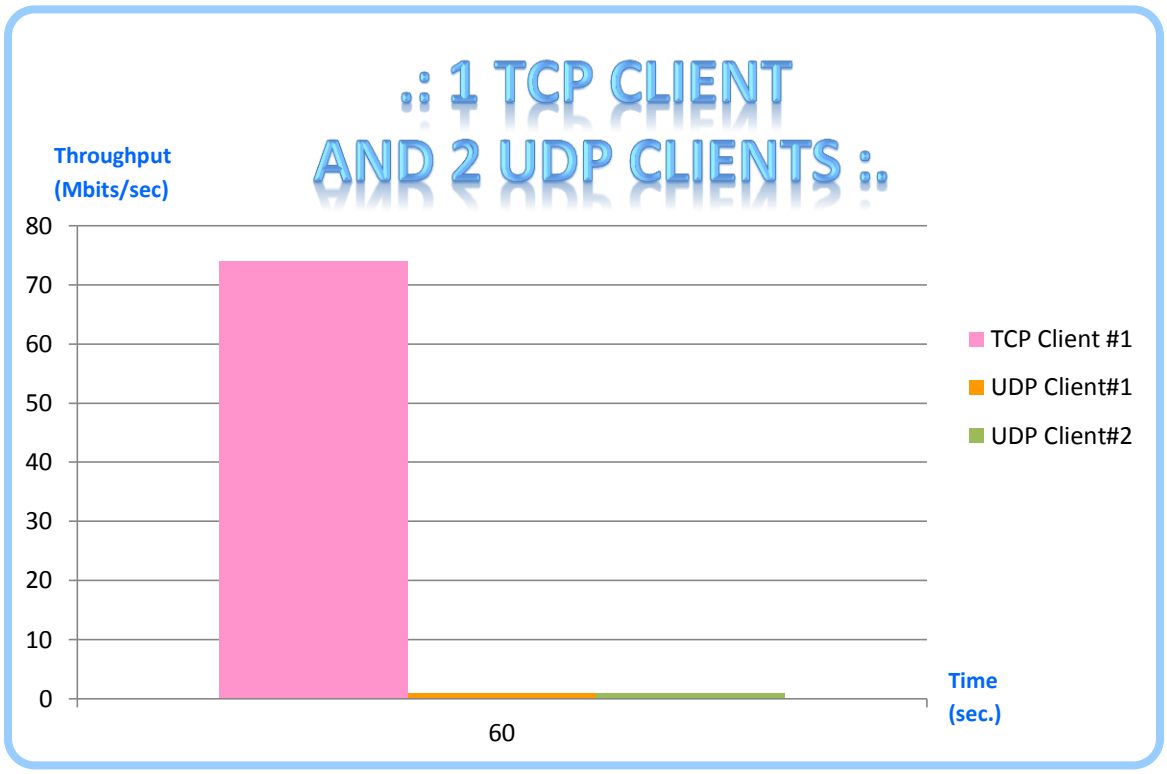
2 TCP clients and 1 UDP client (in table 12)



3 TCP clients and 1 UDP client (in table 13)



1 TCP Client and 2 UDP Clients (in table 14)



...Spread Sheet...

1.) Single TCP session; varying application/protocol parameters.

** TCP receive buffer/window size(-w in iperf) with RTT 0.637 ms

Table 1

Window Size		Transfer		Bandwidth		BDP	
0.124	KBytes	24.1	Mbytes	5.31	Mbits/sec	7.9625	KBytes
0.75	KBytes	187	Mbytes	32.2	Mbits/sec	7.9625	KBytes
2	KBytes	205	Mbytes	57.3	Mbits/sec	7.9625	KBytes
4	KBytes	310	Mbytes	86.8	Mbits/sec	7.9625	KBytes
5.5	KBytes	336	Mbytes	93.6	Mbits/sec	7.9625	KBytes
15	KBytes	338	Mbytes	93.6	Mbits/sec	7.9625	KBytes
55	KBytes	338	Mbytes	93.6	Mbits/sec	7.9625	KBytes
65	KBytes	337	Mbytes	93.6	Mbits/sec	7.9625	KBytes
75	KBytes	338	Mbytes	93.7	Mbits/sec	7.9625	KBytes
85	KBytes	337	Mbytes	93.7	Mbits/sec	7.9625	KBytes
95	KBytes	337	Mbytes	93.7	Mbits/sec	7.9625	KBytes
105	KBytes	338	Mbytes	93.7	Mbits/sec	7.9625	KBytes
115	KBytes	338	Mbytes	93.7	Mbits/sec	7.9625	KBytes
123	KBytes	337	Mbytes	93.7	Mbits/sec	7.9625	KBytes
124	KBytes	338	Mbytes	93.7	Mbits/sec	7.9625	KBytes
125	KBytes	338	Mbytes	93.8	Mbits/sec	7.9625	KBytes
126	KBytes	338	Mbytes	93.6	Mbits/sec	7.9625	KBytes
127	KBytes	337	Mbytes	93.6	Mbits/sec	7.9625	KBytes
135	KBytes	337	Mbytes	93.6	Mbits/sec	7.9625	KBytes
145	KBytes	338	Mbytes	93.7	Mbits/sec	7.9625	KBytes
150	KBytes	338	Mbytes	93.6	Mbits/sec	7.9625	KBytes
1,000	KBytes	338	Mbytes	93.8	Mbits/sec	7.9625	KBytes

** Window Size with RTT 40.634

Table 2

Window Size		Transfer		Bandwidth		BDP	
10	KBytes	8.14	Mbytes	2.27	Mbits/sec	507.925	KBytes
20	KBytes	20.1	Mbytes	5.16	Mbits/sec	507.925	KBytes
55	KBytes	54.2	Mbytes	15.1	Mbits/sec	507.925	KBytes
100	KBytes	88.7	Mbytes	24.7	Mbits/sec	507.925	KBytes
105	KBytes	90.3	Mbytes	25.1	Mbits/sec	507.925	KBytes
150	KBytes	90.3	Mbytes	25.1	Mbits/sec	507.925	KBytes
200	KBytes	91.3	Mbytes	25.4	Mbits/sec	507.925	KBytes
250	KBytes	91.3	Mbytes	25.5	Mbits/sec	507.925	KBytes
300	KBytes	91.4	Mbytes	25.4	Mbits/sec	507.925	KBytes

** Window Size with different delay

Table 3

Window Size		Transfer		Bandwidth		Delay	
125	KBytes	91.8	Mbytes	25.6	Mbits/sec	40	ms
125	KBytes	73.6	Mbytes	24.5	Mbits/sec	50	ms
125	KBytes	61.5	Mbytes	17.2	Mbits/sec	60	ms
125	KBytes	52.7	Mbytes	14.7	Mbits/sec	70	ms
125	KBytes	46.3	Mbytes	12.9	Mbits/sec	80	ms
125	KBytes	41.6	Mbytes	11.6	Mbits/sec	90	ms
125	KBytes	22.6	Mbytes	6.29	Mbits/sec	200	ms
125	KBytes	9.41	Mbytes	2.62	Mbits/sec	500	ms
125	KBytes	4.82	Mbytes	1.28	Mbits/sec	1000	ms
125	KBytes	0.416	Mbytes	0.114	Mbits/sec	5000	ms
125	KBytes	0.016	Mbytes	0.001	Mbits/sec	10000	ms
125	KBytes	Connection timeout		Connection timeout		30000	ms

Window Size		Delay		RTT		BDP	
125	KBytes	40	ms	40	ms	507.5	KBytes
125	KBytes	50	ms	50.7	ms	630	KBytes
125	KBytes	60	ms	60.6	ms	757.5	KBytes
125	KBytes	70	ms	70.5	ms	881.25	KBytes
125	KBytes	80	ms	80.7	ms	1.009	MBytes
125	KBytes	90	ms	90.6	ms	1.113	Mbytes
125	KBytes	200	ms	201.1	ms	2.514	Mbytes
125	KBytes	500	ms	502.3	ms	6.279	Mbytes
125	KBytes	1000	ms	1004.6	ms	12.558	Mbytes
125	KBytes	5000	ms	5007.3	ms	62.591	Mbytes
125	KBytes	10000	ms	10009.7	ms	125.121	Mbytes
125	KBytes	30000	ms	30012.4	ms	375.155	Mbytes

** Length of data written/read by application (-l in iperf) (default 8 kb)

Table 4

Length		Transfer		Bandwidth	
50	KBytes	338	Mbytes	93.6	Mbits/sec
100	KBytes	338	Mbytes	93.6	Mbits/sec
150	KBytes	338	Mbytes	93.7	Mbits/sec
200	KBytes	338	Mbytes	93.7	Mbits/sec
250	KBytes	338	Mbytes	93.7	Mbits/sec
300	KBytes	338	Mbytes	93.6	Mbits/sec
350	KBytes	338	Mbytes	93.7	Mbits/sec
400	KBytes	337	Mbytes	93.7	Mbits/sec
450	KBytes	338	Mbytes	93.7	Mbits/sec
500	KBytes	337	Mbytes	93.6	Mbits/sec
1,000	KBytes	338	Mbytes	93.7	Mbits/sec

2.) Single TCP session; varying network/link conditions.

** Link data rate

Table 5

Data rate (TBF)		Transfer		Bandwidth		Efficiency
50	Kbit	184	Kbytes	44.7	Kbits/sec	89.4 %
100	Kbit	360	Kbytes	92	Kbits/sec	92 %
150	Kbit	440	Kbytes	135.9	Kbits/sec	90.6 %
200	Kbit	464	Kbytes	191.4	Kbits/sec	95.7 %
300	Kbit	496	Kbytes	289	Kbits/sec	96.33 %
500	Kbit	496	Kbytes	430	Kbits/sec	86 %
1000	Kbit	488	Kbytes	970	Kbits/sec	97 %
2000	Kbit	512	Kbytes	1.88	Mbits/sec	94 %
5000	Kbit	512	Kbytes	4.96	Mbits/sec	99.2 %
10000	Kbit	512	Kbytes	9.32	Mbits/sec	93.2 %

** Packet drop rate

Table 6

Packet Loss	Transfer		Bandwidth	
1 %	310	Mbytes	86.6	Mbits/sec
2 %	203	Mbytes	56.8	Mbits/sec
3 %	132	Mbytes	36.6	Mbits/sec
5 %	51.4	Mbytes	14.2	Mbits/sec
10 %	11.1	Mbytes	3.11	Mbits/sec
50 %	24	Mbytes	1.82	Mbits/sec
60 %	64	Kbytes	15	Kbits/sec
65 %	Time Out			
70 %	Time Out			
80 %	Time Out			
100 %	Time Out			

** Link delay

Table 7

Delay (ms)	Range(ms)	Transfer		Bandwidth	
100	10	235	Mbytes	65.3	Mbits/sec
100	20	242	Mbytes	67.8	Mbits/sec
100	30	245	Mbytes	67.1	Mbits/sec
150	10	181	Mbytes	50.3	Mbits/sec
200	10	160	Mbytes	44.5	Mbits/sec
250	10	173	Mbytes	48.3	Mbits/sec
		183	Mbytes	51.1	Mbits/sec
		177	Mbytes	49.3	Mbits/sec
		180	Mbytes	50.3	Mbits/sec
300	10	143	Mbytes	39.9	Mbits/sec
350	10	118	Mbytes	33	Mbits/sec
400	10	95.7	Mbytes	26.6	Mbits/sec
700	10	38.3	Mbytes	10.6	Mbits/sec

3.) Multiple TCP sessions.

** 2 TCP clients

Table 8

Time	client	Transfer		Bandwidth	
60 sec.	client #1	317	Mbytes	51.3	Mbits/sec
	client #2	320	Mbytes	43.5	Mbits/sec
SUM		691	Mbytes	94.1	Mbits/sec

** 3 TCP clients

Table 9

Time	client	Transfer		Bandwidth	
60 sec.	client #1	296	Mbytes	40.8	Mbits/sec
	client #2	217	Mbytes	29	Mbits/sec
	client #3	211	Mbytes	27.1	Mbits/sec
SUM		724	Mbytes	94.1	Mbits/sec

** 4 TCP clients

Table 10

Time	client	Transfer		Bandwidth	
60 sec.	client #1	209	Mbytes	28.7	Mbits/sec
	client #2	174	Mbytes	23.4	Mbits/sec
	client #3	168	Mbytes	22.2	Mbits/sec
	client #4	177	Mbytes	22.9	Mbits/sec
SUM		728	Mbytes	94.1	Mbits/sec

4.) Single/Multiple TCP sessions in presence of UDP sessions.

** 1 TCP clients and 1 UDP client

Table 11

Time		Transfer		Bandwidth	
TCP					
60	sec.	668	Mbytes	93	Mbits/sec
UDP					
60	sec.	7.5	Mbytes	1.05	Mbits/sec
TCP with bandwidth 0.5					
60	sec.	663	Mbytes	92.3	Mbits/sec
UDP with bandwidth 0.5					
60	sec.	7.5	Mbytes	1.05	Mbits/sec
TCP with bandwidth 50					
60	sec.	675	Mbytes	94.1	Mbits/sec
UDP with bandwidth 50					
60	sec.	87.6	Mbytes	11.8	Kbits/sec
TCP with bandwidth 100					
60	sec.	676	Mbytes	94.1	Mbits/sec
UDP with bandwidth 100					
60	sec.	87.6	Mbytes	11.8	Mbits/sec

** 2 TCP clients and 1 UDP client

Table 12

Time	client	Transfer		Bandwidth	
2 TCP					
60 sec.	client#1	376	Mbytes	52	Mbits/sec
	client#2	317	Mbytes	41	Mbits/sec
SUM		693	Mbytes	89.6	Mbits/sec
300 sec.	client#1	1.56	Gbytes	44.6	Mbits/sec
	client#2	1.51	Gbytes	42.8	Mbits/sec
SUM		3.07	Gbytes	87.1	Mbits/sec
1 UDP					
60 sec.	client#1	7.5	Mbytes	1.05	Mbits/sec
300 sec.	client#2	37.5	Mbytes	1.05	Mbits/sec

** 3 TCP clients and 1 UDP client

Table 13

Number of client/Time		Transfer		Bandwidth	
3 TCP					
60 sec.	client#1	243	Mbytes	33.6	Mbits/sec
	client#2	195	Mbytes	26.3	Mbits/sec
	client#3	202	Mbytes	26.6	Mbits/sec
SUM		641	Mbytes	84.3	Mbits/sec
1 UDP					
60 sec.	client#1	7.5	Mbytes	1.05	Mbits/sec

** 1 TCP client and 2 UDP clients

Table 14

Number of client/Time		Transfer		Bandwidth	
1 TCP					
60 sec.	client#1	532	Mbytes	74	Mbits/sec
1 UDP					
60 sec.	client#1	7.5	Mbytes	1.05	Mbits/sec
	client#2	7.5	Mbytes	1.02	Mbits/sec
SUM		15	Mbytes	2.05	Mbits/sec