Assignment 2

Present to

Dr.Steven Gordon

Prepare By

Group 6

Mr. Methee Choppatanar	ID 5122780752
Mr. Suravit Jitchikan	ID 5122781347
Mr. Ariyachai Chaimanat	ID 5122790389
Mr. Asanee Plienpun	ID 5122791098

Sirindhorn International Institute of Technology 2010

Objective

To understand the TCP and UDP how it work?

Specifications

- Server
  - VAIO VGN-TX16SP
  - Network interface 10/100Mbps Fast Ethernet
  - Ubuntu 10.10 Linux kernel 2.6.35
- Client
  - VAIO VGN-CR320E
  - Realtek RTL8101E Family PCI-E Fast Ethernet NIC
  - Ubuntu 10.10 Linux kernel 2.6.35
- Cable
  - Cross over cable 100 mbps
- Software
  - Iperf
  - tc
  - iptables

Task 1 : Single TCP session varying application/protocol parameters

1.1 Window Size

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- *Change window size at server*
  - Server `iperf -s -p 50123 -w xxxxx`
  - Client: `iperf -c 1.1.1.3 -p 50123 -t 30`
  - We start window size from 100 to 100000
  - Collect and consider the throughput that has changed
- *Change window size at client*
  - Server: `iperf -s -p 50123`
  - Client: `iperf -c 1.1.1.3 -p 50123 -t 30 –w xxxxx`
  - We start window size from 100 to 100000
  - Collect and consider the throughput that has changed

Result

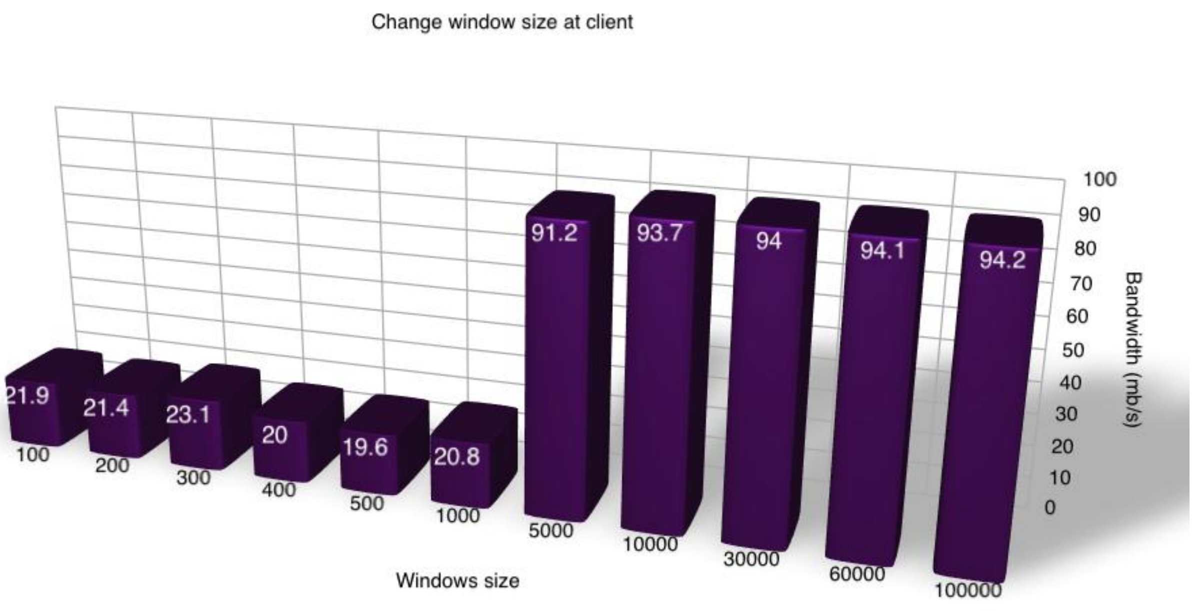

Figure 1

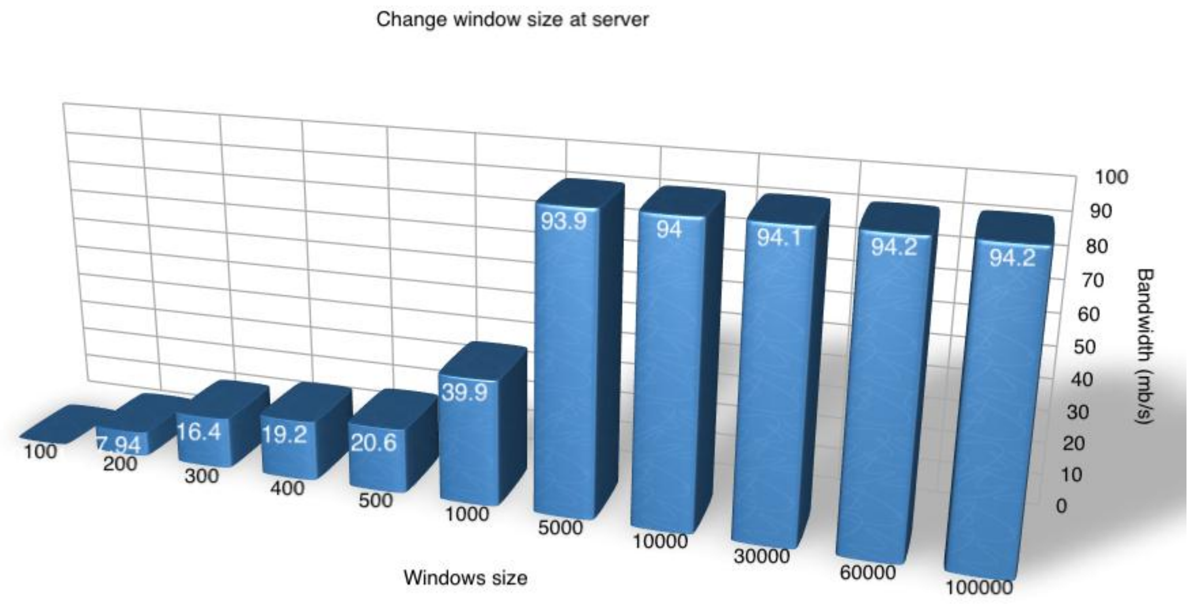

Figure 2

Discussion

In this task, We want to find the maximum throughput(or Bandwidth) by sending many of the different window size to measures of the maximum throughput that was send from the different of packet size and focus on the send and the receive traffic flow buffers.

In this scenario, the client try to allocate the buffer sizes by sending the window size to told the available buffer size of itself to the server, the window size are refer as the receiver’s buffer size.

First, We initialize the window size of 100 and we can see that the bandwidth result is equals to 21.9Mb/s and after that we increased the another five window size which are 200,300,400,500 and 1000 that make the throughputs quite changed a little bit around 2-3Mb/s from 20-23.1 Mb/s. It’s seemed that the throughputs are constant but when we change the size of window which is 5000. We can see that the throughputs also increased. From this result we have to find exactly the constant throughput by increasing the window size to 10000, 30000, 60000 and 100000 and we get the result that very constant of the maximum throughput changed around 1-3Mb/s between 91.2Mb/s to 94.2Mb/s.

From this result, we can see that the maximum throughput is very constant because from the increased of window size from 30000 to 60000 and 60000 to 100000. The maximum throughput has a quite change around 0.2Mb/s (94Mb/s to 94.2Mb/s).

From the Figure 1 and 2, we can refer that the more window size increasing, the more bandwidth increasing also. The main reason is that window size control amount of data waiting in buffer. When the packet is received by node, it is not necessary sent out from buffer immediately. It is window size that determines this process. Window size is a maximum size of data that can be handled before it is passed to the application process. Window size is controlled by flow control of TCP. When the window size is increased, there will be more space for handle incoming data. This will cause more throughputs.

The other reason is efficiency of network. When increasing window size, there are more packets sending because there will faster received ACK from another node. Then this will help to optimize network that will increasing throughput. Consequently, we can conclude that the increasing of window size will get the result of the maximum throughput.

1.2 Length

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- *Change length at server*
  - Server `iperf -s -p 50123 -l xxxxx`
  - Client: `iperf -c 1.1.1.3 -p 50123 -t 30`
  - We start length from 1,2,4,8,16 … 1024
  - Collect and consider the throughput that changed
- *Change length at client*
  - Server: `iperf -s -p 50123`
  - Client: `iperf -c 1.1.1.3 -p 50123 -t 30 –l xxxxx`
  - We start length from 1,2,4,8,16 … 1024
  - Collect and consider the throughput that changed

Result

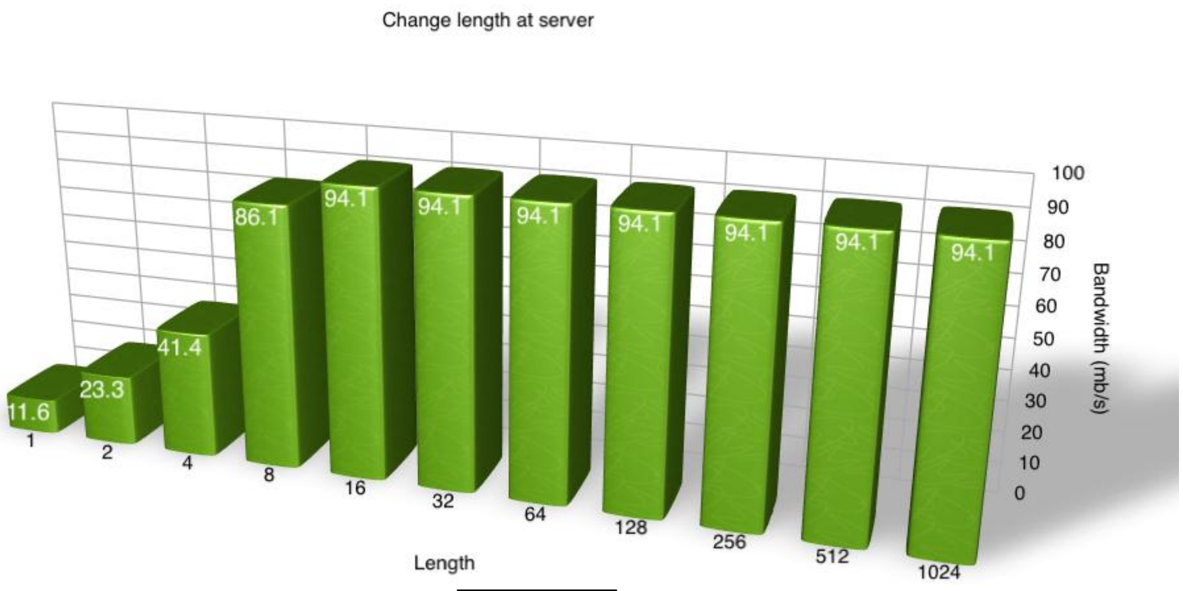

Figure 3

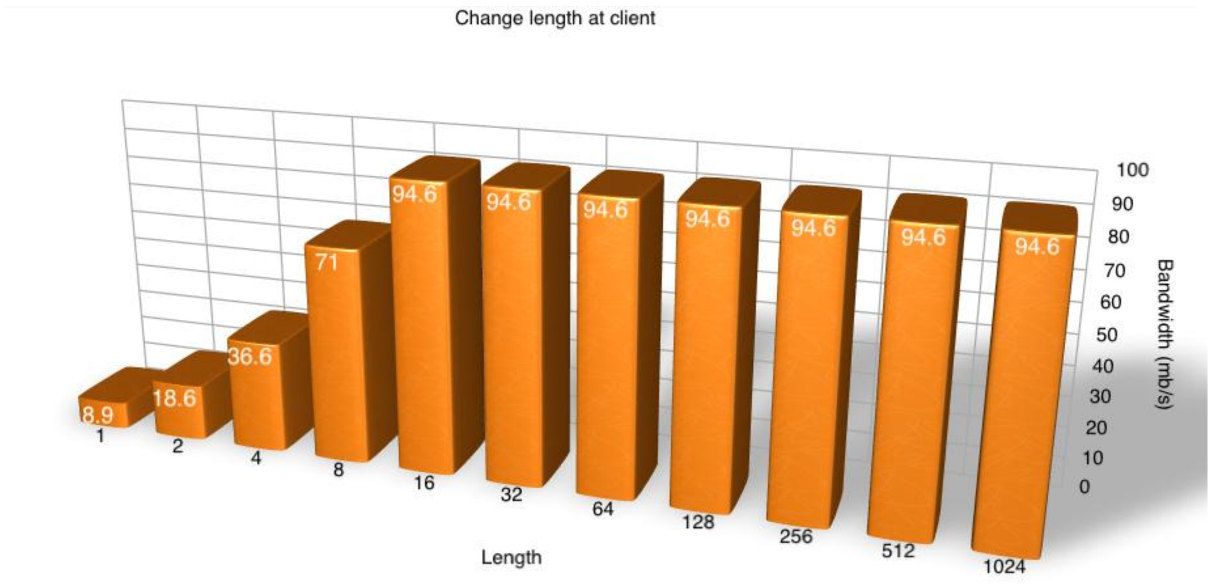

Figure 4

Discussion

From the figure 3, we can see that at the server's computer from the size of length in Kbytes unit from 1, 2, 4 and 8 the result in the throughput are not constant which is 11.6, 23.3, 41.4, 86.1Mbits/s means that we need to increasing the size of length to get the maximum throughput and when we increase the value of length size. The Throughput is still not constant which is increased from 86.1Mbits/s to 94.1Mbits/s and when we keep increasing the size of length by multiply by 2. We still get the result same as the client's computer in which the result in the throughput are stayed constant.

From the figure 4, we can refer that the increasing in the size of length still increased the throughput to find the maximum throughput. First, we initialize the size of length to 1kbyte and we can see that the throughput is equals to 8.9Mbits/s and we increase the length to see that the maximum throughput is constant by changing the size of length to 2, 4, 8, 16 but the result of the throughput are still not constant with 18.6, 36.6, 71 and 94.6Mbits/s. from this result we still need to find the exactly throughput because the results are not constant. And after that when we are increasing the length from 16Kbytes to 32Kbytes. We can see that the amount of the throughput is staying constant which is 94.6 Mbits/s and when we increasing the size of length by multiply by 2 to 64, 128, 256, 512 and 1028. We can get the same result on the throughput of the size of 32 of length. Now we can see that the result of the throughput is constant and never changed.

The main reason that bandwidth is constant because the data length is reach its maximum level.it cannot be more than this. The value is tends to be 94.1 Mbits/s and 94.6 Mbits/s respectively because our experiment is using Ethernet LAN cable as a medium. It will try to reach 100 Mbits/s. it is properties of LAN cable that has bandwidth about this.

Task 2 : : Single TCP session with varied network/link parameters.

2.1 Link data rate

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- At server
  - Server : `iperf -s -p 50123`
- At client
  - Set up rate value
    - Client : `sudo tc qdisc add dev eth0 root tbf rate xxxkbit latency 50ms burst 5kb`
  - Change rate value
    - Client: `sudo tc qdisc replace dev eth0 root tbf rate xxxkbit latency 50ms burst 5kb`
  - client : `iperf -c 1.1.1.3 -p 50123 -t 30`
  - Input rate from 50kbit to 10000kbit
  - Collect bandwidth

Result

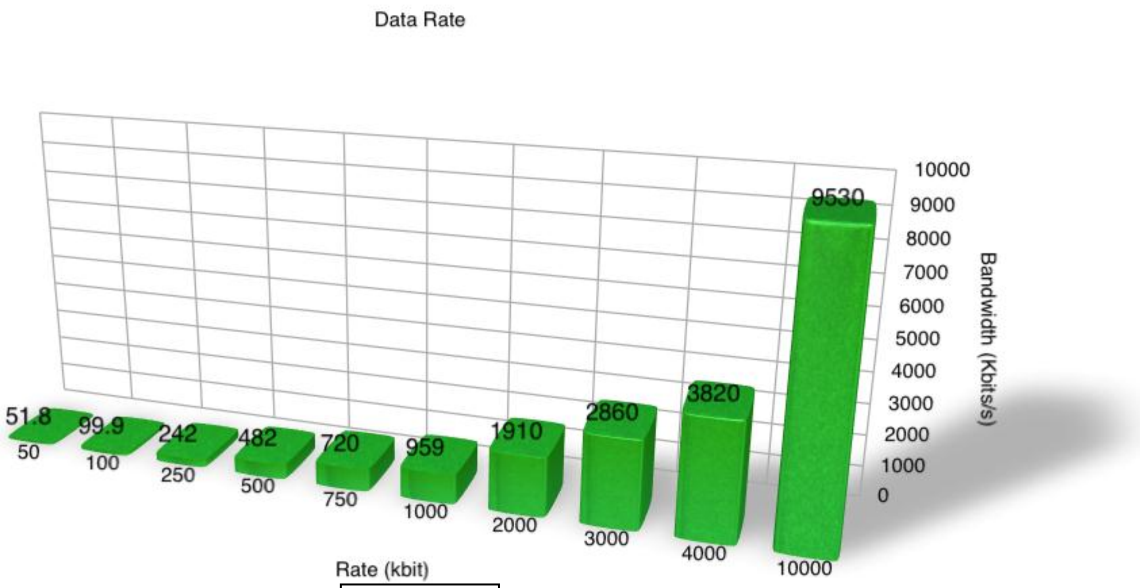

Figure 5

Discussion

From the Figure 5, we can refer that bandwidth is increasing when total amount of token bucket filter is increasing. First, we try to see when using TBF at only 50 kbit, amount of Bandwidth is equals to 51.8 Kbits/sec. we can see when using TBF at 100 kbit, amount of Bandwidth is equals to 99.9 Kbits/sec. Then we can see when using TBF at 10000 kbit, amount of Bandwidth is equals to 9.53 Mbits/sec. We can imply that the specified TBF rate has controlled bandwidth. Bandwidth is always nearly too specified TBF rate.

The Token Bucket Filter is a queuing discipline for traffic control. It will limited connection to sending out data at the rate that is not more than the assigned value , even there are a gap available in network. It may allow short bursts in excess of the assigned value. TBF create a buffer as bucket and some virtual pieces of information as token at a specific rate (token rate).Token that want to be sent will go at steady rate. When buffer is full under the configured rate, packets need to queue up and wait until the buffer has sent a first packet in a queue. This is when we can see the limited rate of TBF.

The reason that received bandwidth is always nearly to specify TBF rate is the actual bandwidth is much more than the specified TBF rate. The bandwidth tends to go to the actual rate. By far from the actual rate, bandwidth tends to go to the limited value, that is TBF, or the biggest value that it can possible has. There are still some little bit different with TBF rate and received rate because there are some headers of data including.

2.2 Link delay

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- At server
  - Server : `iperf -s -p 50123`
- At client
  - Client : `sudo tc qdisc add dev eth0 root netem delay xxms 10ms`
  - Client : `iperf -c 1.1.1.3 -p 50123 -t 30`
  - We change delay from 1 to 5000
  - Collect bandwidth that changed

Result

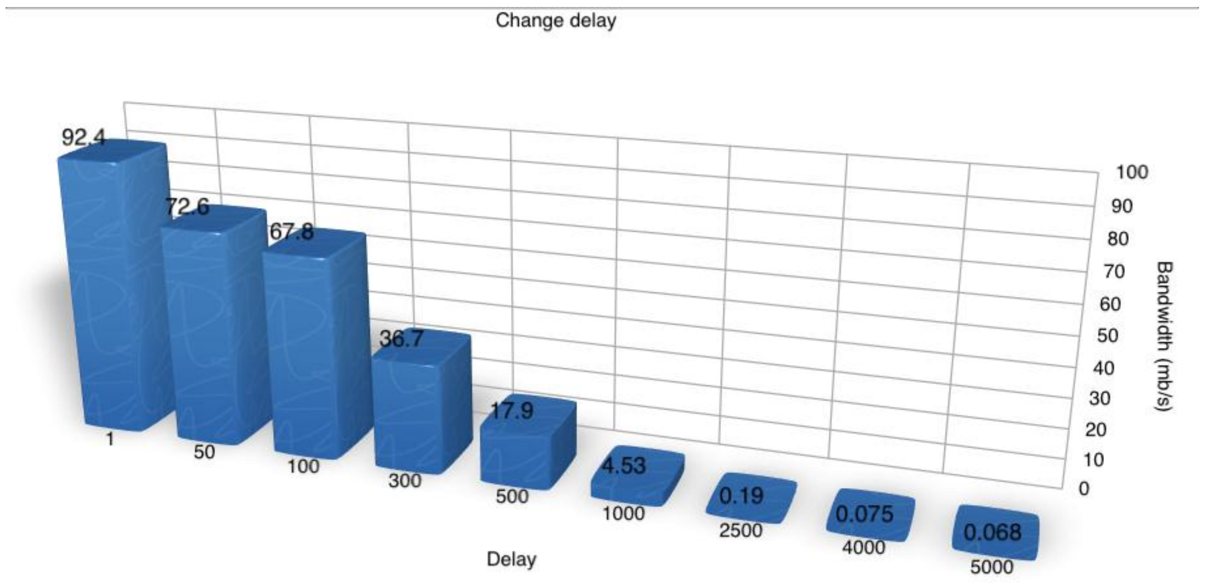

Figure 6

Discussion

From the graph above, First, we try to see when using 1ms delay, amount of Bandwidth is equals to 92.4Mbits/sec. We can see when using 50 ms delay, amount of Bandwidth is equals to 72.6
Mbits/sec. Then we can see when using 5000 ms delay, amount of Bandwidth is equals to 68.4 kb/s.

From this experiment, we can see that at the first time we initialize the delay for 1ms and the Bandwidth result is 92.4Mbits/s .but when we increasing the amount of delay value, we can see that if the value of delay is increased, the Bandwidth is decreased in every added delay value. Consequently, we can conclude that the delay is affect to the Bandwidth to measure the TCP performance.

Then, we will compare about different delay time. From the Figure xxx, we can see that the more delay increasing, the less bandwidth system get. There are 2 factors that essentially effect throughput. First, there has longer waiting time and round trip time. With longer waiting time, this will caused every transfer has more round trip time. Receiver node has to wait for more time.it may be too long for receiver node and excess ACK times out. The receiver node may be considered this packet as packet loss whether it is or not. Then there will be a retransmission and the retransmission may be happened again and again because of the same reason. This retransmission will cause reducing of throughput.

2.3 Packet Drop

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- At server
  - Input probability value from 0.01 to 0.5 in this command
  - Server: `sudo iptables -A INPUT -m statistic --mode random --probability xxx -j DROP`
  - Server : `iperf -s -p 50123`
- At client
  - In case of 30 sec
    - Client : `iperf -c 1.1.1.3 -p 50123 -t 30`
  - In case of 60 sec
    - client : `iperf -c 1.1.1.3 -p 50123 -t 60`
- Collect bandwidth between 2 cases

Result

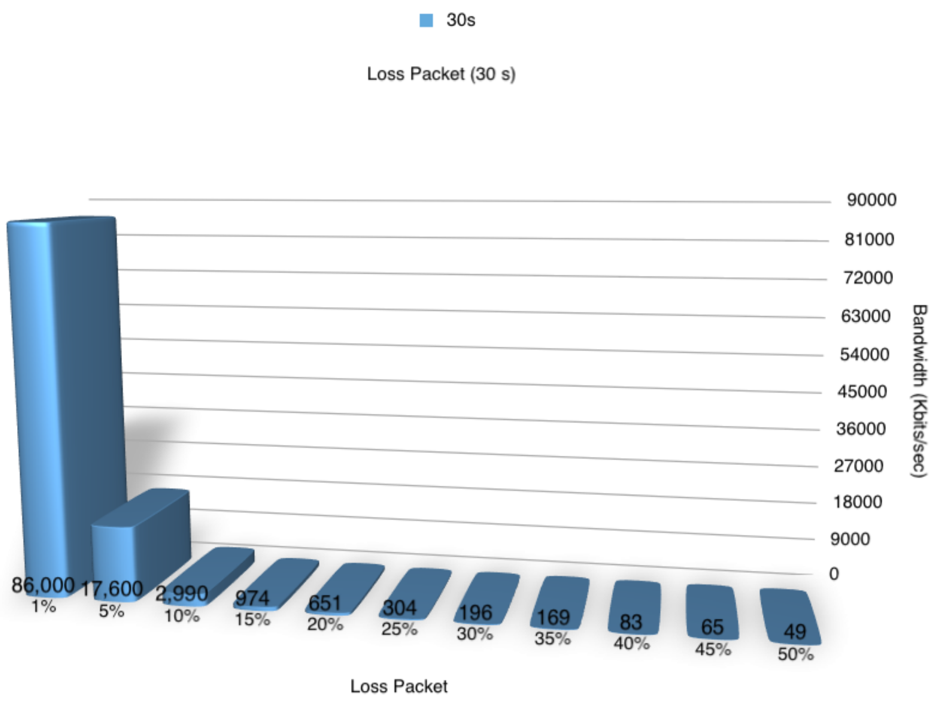

Figure 7

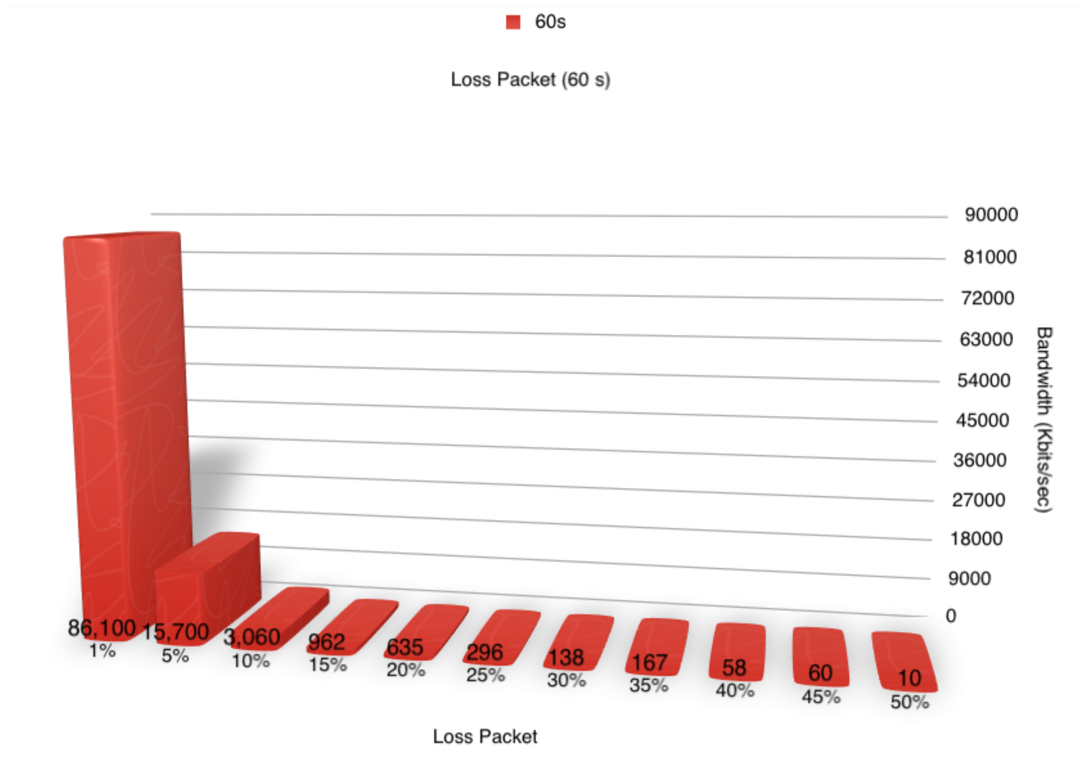

Figure 8

Discussion

From the Figure 7 and 8, first, we try to see when using loss rate at 1%, amount of Bandwidth is equals to 86.0 Mbits/sec. we can see when using rate at 5%, amount of Bandwidth is equals to 17.6 Mbits/sec. Then we can see when using rate at 50%, amount of Bandwidth is equals to 48.9 Kbits/sec.

First, we can refer that the more percentage of loss rate, the less amount of Bandwidth. The main reason is the rate of loss packet determine amount of data exchange. If there are more percentage of loss rate, it means that more packet are missing and it will cause retransmission and inefficient of network. Bandwidth is just like data rate. It depends on how many packets send and receive. With more packet loss, bandwidth will be less.

Next, we will compare about different waiting time. From the Figure 7 and 8, we can see that within the same loss rate, bandwidth of 30 sec waiting time is, mostly, more than bandwidth of 60 sec waiting time. There are 2 factors that essentially effect throughput. First, it is periods that are not equal. 30 sec waiting time has more bandwidth because the longer time experiment takes the more probabilities to have a packet loss. These probabilities of packet loss effect clearly when there are large different period to see. Second, it is an ACK times out. With longer waiting, receiver node has to wait for more time.it may be too long for receiver node and excess ACK times out. The receiver node may be considered this packet as packet loss whether it is or not. Then there will be a retransmission and the retransmission may be happened again and again because of the same reason. This retransmission will cause reducing of throughput.

Special Task: Relationship between protocol parameters and network parameters

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- Server ping to client
  - Ping 1.1.1.2
  - AVG Rtt = 80.823
- Find Data Rate
  - Client : `sudo tc qdisc replace dev eth0 root netem delay 80ms 10ms`
  - Client : `sudo tc qdisc add dev eth0 root tbf rate 100000kbit latency 50ms burst 5kb`
- BDP = AVG Rtt * Data Rate
- Change WD size At client
  - Server : `iperf -s -p 50123`
  - Client : `iperf -c 1.1.1.3 -p 50123 -t 30 -w xxxx`
- Change WD size At server
  - Server : `iperf -s -p 50123 -w xxxx`
  - Client : `iperf -c 1.1.1.3 -p 50123 -t 30`
- Collect bandwidth

Result

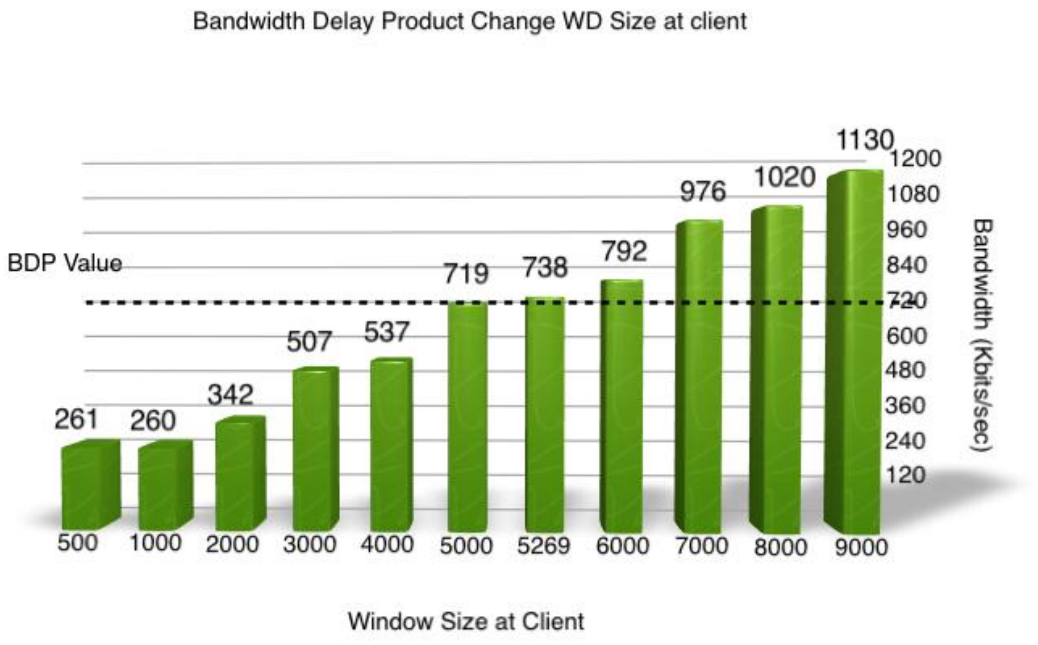

Figure 9

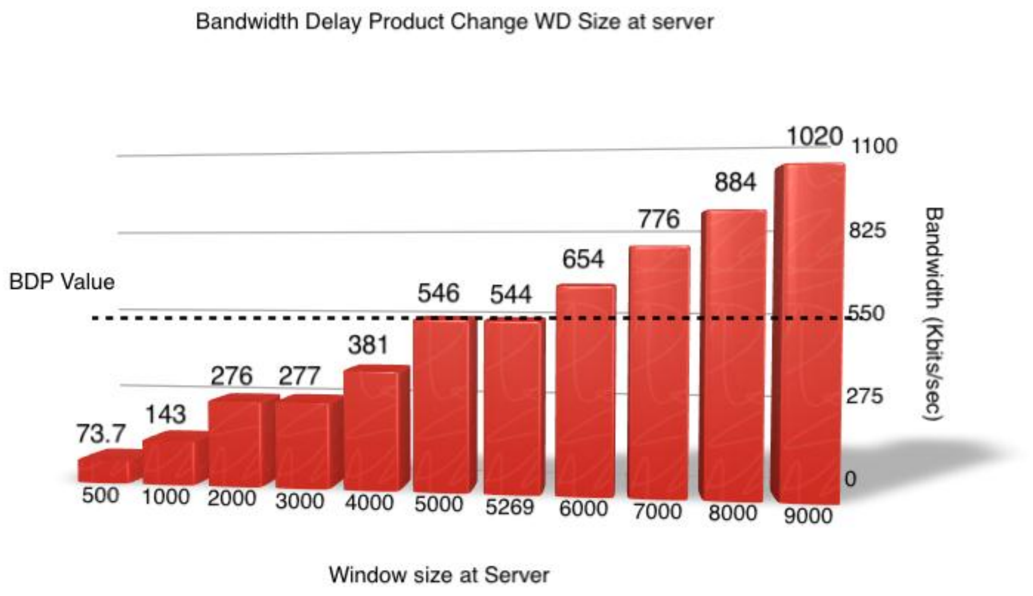

Figure 10

Discussion

From the Figure 10, we can refer that Bandwidth Delay Product of server side is about 5269.6 Kbits. It comes from average round trip time (ms) * Max data rate with delay 80ms (Mbits/sec). First, we try to see when using 500 of window size, amount of Bandwidth is equals to 73.7 Kbits/sec. we can see when using 5269 of window size, and amount of Bandwidth is equals to 544 Kbits/sec. Then we can see when using 9000 of window size, amount of Bandwidth is equals to 1.02 Mbits/sec.

Bandwidth Delay Product is for measure how many data that can be transfer in the network in the short period. It means that throughput depends on BDP. BDP has an important role in high speed network and optimizing system and TCP to get more efficient.

We can imply from the Figure 10 that the more window size at server side, the more bandwidth that network receive. The main reason is efficiency of network is limited by window size. Every time we increase window size, bandwidth will increase together. To maximize usage of network, we have to tune up system by using optimal TCP window size. BDP helps to get this value because BDP is amounting of data that sender must send before first packet arriving at the receiver.

Task 3: Multiple TCP sessions

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- At server
  - Run server
    - Server : `iperf -s -p 50123`
- At client
  - Run client 60sec and specific number of port
    - Client : `iperf -c 1.1.1.3 -p 50123 -t 60 -P xxxx`
- Collect sum bandwidth and each session

Result

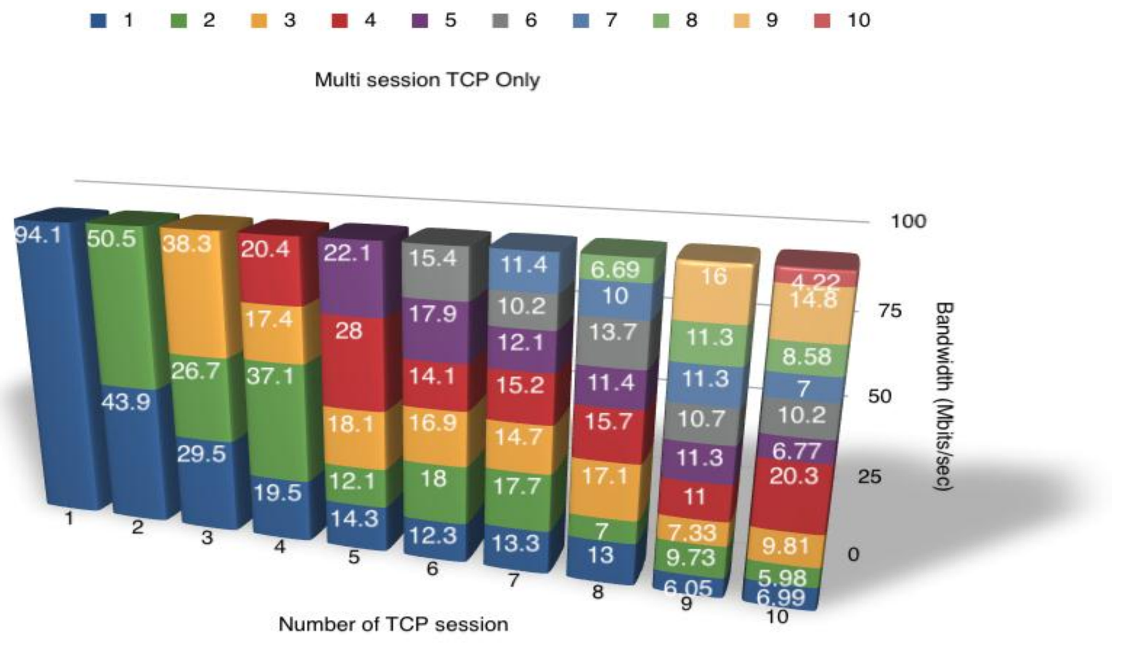

Figure 11

Discussion

From the Figure 11, we can refer that Average sum of Bandwidth of TCP session is about 94.1 Mbits/sec. First, we try to see when using only 1 of TCP session; amount of Bandwidth is equals to 94.4 Mbits/sec. We can see when using 2 of TCP session, amount of Bandwidth is equals to 94.1 Mbits/sec. Then we can see when using 10 of TCP session, amount of Bandwidth is equals to 94.1 Mbits/sec.

First, we look at Average Bandwidth for each TCP session with different Number of TCP session. It can be implying that more Number of TCP session in experiment, less Average Bandwidth for each TCP session. The main reason is that TCP has to manage limited resources under some condition. Between the experiments, bandwidth is stable. We can consider bandwidth as a resource. This resource is come from lower layer, like internet layer and network interface, and arrives at TCP layer. TCP has some standard to allocate limited resource among different session fairly. It has to distribute bandwidth to every session. Then we can see that even average bandwidth of each TCP session is decreasing when increasing Number of TCP session, sum of every TCP session is always equal to 94.4 Mbits/sec.

Second, Even though we change amount of TCP session, it did not change amount of bandwidth. Bandwidth will stable at 94.4 Mbits/sec for all the time of experiment. The main reason is TCP does not change any bandwidth from lower layer. TCP is just receiving from them and manage resource to application.

Task 4: Single/Multiple TCP sessions in presence of UDP sessions

Procedure and Command

- We set up the ip of server and client
  - Server : 1.1.1.3
  - Client : 1.1.1.2
  - Port : 50123
- At server
  - Run 2 servers
    - Server 1 : `iperf -s -p 50123`
    - Server 2 : `iperf -s -u -p 50123`
- At client
  - Run 2 clients
    - Client1 : `iperf -c 1.1.1.3 -p 50123 -t 60 -P xxxx`
    - Client2 : `iperf -c 1.1.1.3 -u -p 50123 -t 60`
- Collect Bandwidth of TCP
- Collect Bandwidth of UDP
- Collect Packet loss of UDP

Result

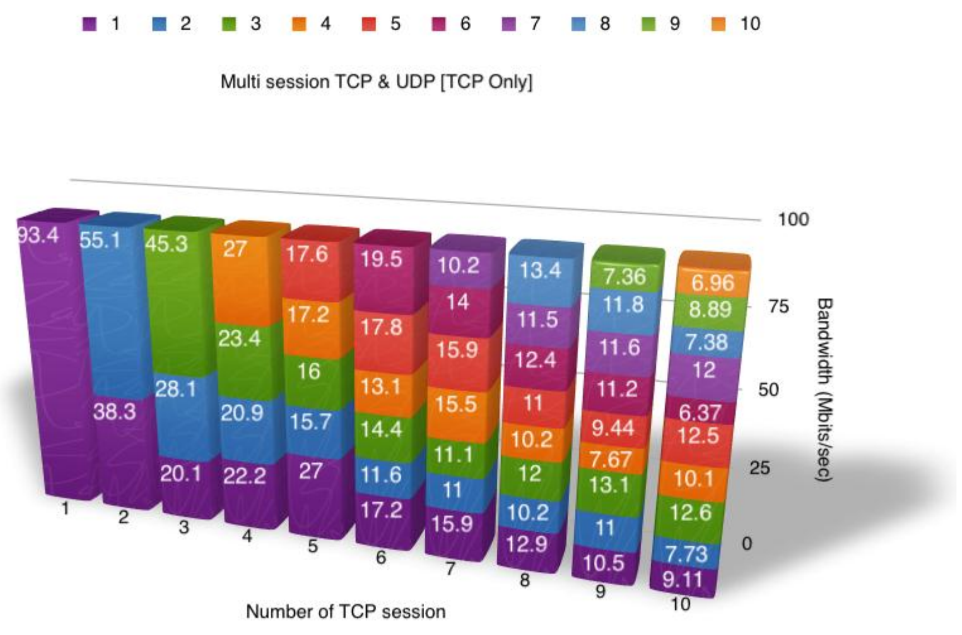

Figure 12

Discussion

From the Figure 12, we can refer that Average sum of Bandwidth of TCP session is about 93.4 Mbits/sec. First, we try to see when using only 1 of TCP session, amount of Bandwidth is equals to 93.4 Mbits/sec. we can see when using 2 of TCP session, and amount of Bandwidth is equals to 93.5 Mbits/sec. Then we can see when using 10 of TCP session, amount of Bandwidth is equals to 93.5 Mbits/sec.

Even though we change amount of TCP session, it did not change amount of bandwidth. Bandwidth will stable at 93.4 Mbits/sec for all the time of experiment. It is always generates definitely throughput for all the time. Bandwidth did not depended on number of TCP session because TCP is in a part of application layer of TCP/IP model but Bandwidth is controlled by lower layer. It means that lower layer will manage throughput and TCP will allocate these resources by its appropriate measure.

We can see that with more number of TCP session, it has less of Bandwidth per each TCP session. Because of sharing between sessions, it has to allocate its limited resources to every session. But if we calculate sum of each session, there will be equals nearly to 93.4 Mbits/sec.

TCP has properties of fairness and distribution. It will get stable bandwidth from lower layer such as internet layer and network interface. It will allocate this bandwidth to every session with its appropriate measure. Every session may get resources equally.

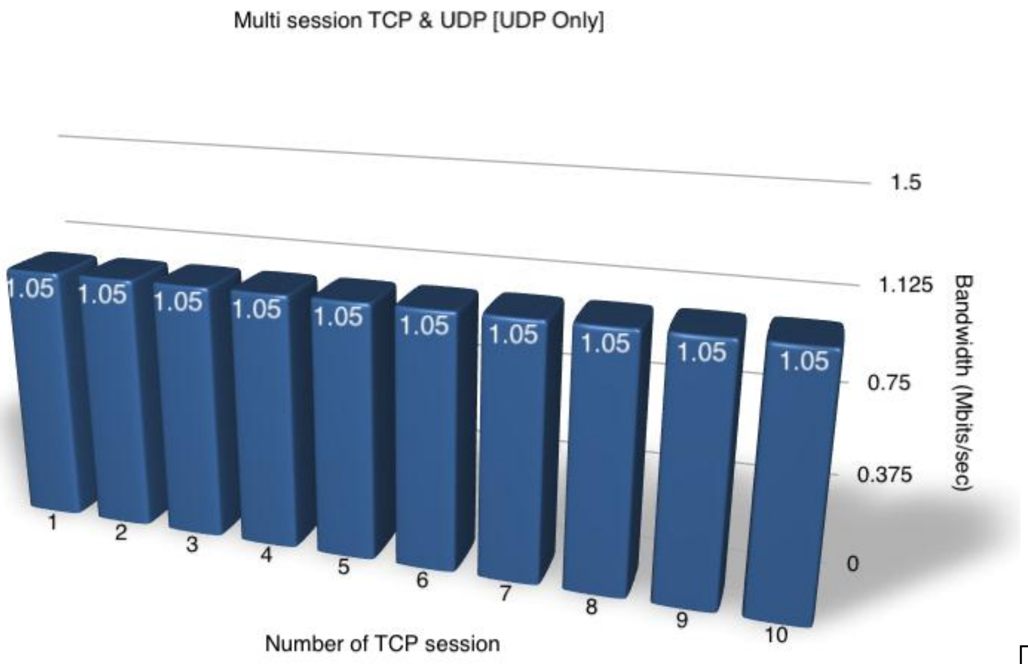

Figure 13

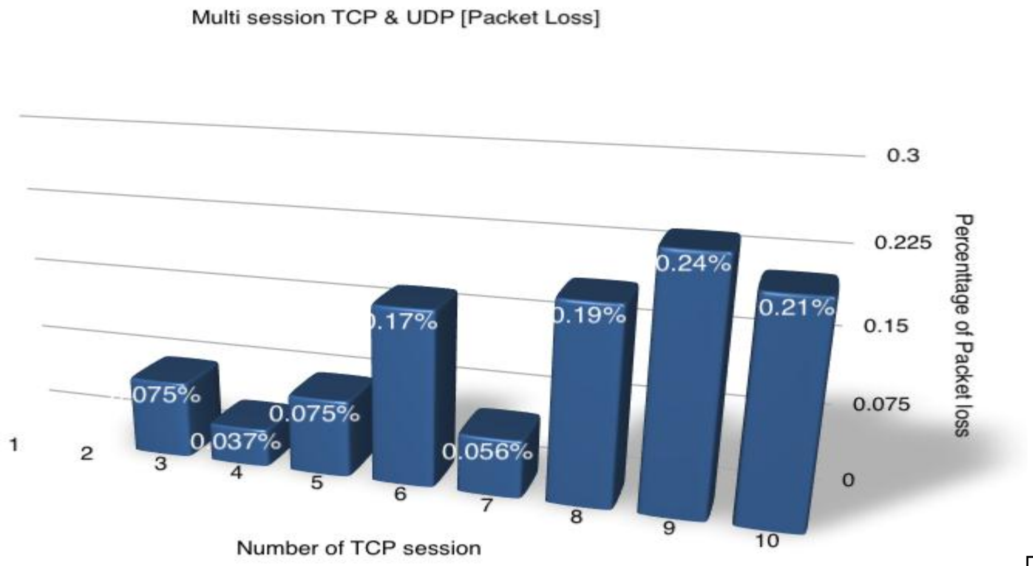

Figure 14

From the Figure 13, we can refer that Average Bandwidth of UDP is about 1.05 Mbits/sec.First, we start looking at when using only 1 of TCP session, amount of Bandwidth is equals to 1.05 Mbits/sec and packet loss is 0%. We see when using 2 of TCP session, amount of Bandwidth is equals to 1.05 Mbits/sec and packet loss is 0.075%. Then we see when using 10 of TCP session, amount of Bandwidth is equals to 1.05 Mbits/sec and packet loss is 0.21%.

Even though we have different numbers of TCP session, the result of average bandwidth of UDP is equal to 1.05 Mbits/sec, average of packet loss of UDP is less than 0.2% and packet loss of UDP is likely to increasing when there are more numbers of TCP sessions. The reason that makes average bandwidth of UDP is equals for every time of experiment is because the value, actually, changed. When we plot our graph in a big scale, like Mbits/sec, we cannot see the differences. But when we plot it again in a small scale, like Kbits/sec or number of packet loss, there will be more precise and clearer.

Then we have to see From the Figure 14, we can see that number of packet loss are increasing when number of TCP session are increasing. We can imply from this statement that number of TCP session has effect to number of UDP packet loss. The reason is with more number of TCP sessions happening on network, there will has more number of packet coming to the node. This will cause buffer to receive more packet and then get congested and then more packet loss happening.

There is one another reason that causes more UDP packet loss. We see that TCP has properties of fairness to manage their packets but UDP packet has no any measure to handle them. It means that TCP will get more priority or better management than UDP packets. Under limited resources, the bandwidth will be allocated by TCP for its packets. It means that TCP packet will get more privileges to resource.

Then UDP will get less. Finally, UDP packet will have trouble with transfer and may have bottleneck. The packet loss will happen.

Conclusion

After we do all of experiment, we have a lot of about transferring data on network testing by Iperf running on Linux-based operating system.

First, we have tested about Single TCP session with varied application/protocol parameters. We have tested in Window Size topic. We have learned that the more window size, the more bandwidth system get. We have tested in Length of Data. We have learned that the more data length for buffer, the more bandwidth server and client received.

Second, we have tested about Single TCP session with varied network/link parameters. We have tested in Link Data Rate. We have learned that the more link data rate we set, the more bandwidth system received. We have tested in Link Delay. We have learned that the more link delay we set, the more bandwidth system received. We have tested in Packet Dropped. We have learned that the more loss percentage we set, the more data loss between network. Then, we have tested about Relationship between protocol parameters and network parameters. We have learned that the more window size at server, the more bandwidth system received.

Next, we have tested about Multiple TCP sessions. We have learned that the more number of sessions in the network, the less bandwidth each session gained.

Finally, we have tested about Single/Multiple TCP sessions in presence of UDP sessions. We have learned that the number of TCP sessions, the more number of packet loss in system.

In conclusion, we have learned what is performance of TCP with multiple factor under different conditions. We have learned that network has a lot of factor to consider for tune up it to be optimized. Under vary condition, we must considered different factors.

Reference

https://lists.linux-foundation.org/pipermail/netem/2009-May/001325.html
http://lartc.org/howto/lartc.qdisc.classless.html
http://opalsoft.net/qos/DS-24.htm
http://linux.die.net/man/8/tc-tbf
http://www.isoc.org/inet97/proceedings/F3/F3_1.HTM
http://www.diffen.com/difference/TCP_vs_UDP#Connection
sd.wareonearth.com/woe/Briefings/tcptune/tsld008.htm
http://www.kehlet.cx/articles/99.html
http://www.speedguide.net/faq_in_q.php?qid=185
http://www.isoc.org/inet97/proceedings/F3/F3_1.HTM
http://www.diffen.com/difference/TCP_vs_UDP#Connection