Improvement Ivestigation of the TCP Algorithms With Avoiding Network Congestion Based on OPNET

Improvement Ivestigation of the TCP Algorithms With Avoiding Network Congestion Based on OPNET

Haeeder M. Noman[1], Ali A. Abdulrazzaq[1], Marwah M. Kareem[2,3] and Adnan Hussein Ali[1]
[1]Technical Instructors Training Institute, Middle Technical University, Baghdad, Iraq
[2]Faculty of Engineering & Built Environment, Universiti Kebangsaan Malaysia, Malaysia
[3]Computer Techniques Engineering, Al-Israa College University, Baghdad, Iraq

Abstract. The aim of this paper is to determine the performance enhancement of the network for eliminating congestion through the use of the benefits of algorithms. These algorithms can be used in hindering the congestion which may occur within the network that relies on a Transmission Control Protocol (TCP) protocol. This paper specifically focuses on demonstrating the function of TCP, and most especially the four (slow start, congestion avoidance, fast retransmit and fast recovery algorithms) that are applied in the control of congestion. In the study, the varying effects of connection link from PPP DS1 to PPP DS3 on the network's performance are investigated.

Keywords: Congestion control, slow start, Congestion avoidance, Fast recovery Connection handshake, Fast retransmit Flow Control, DS3, T1

1. Introduction

The advancement in network technology has further enhanced the flexibility of internet operations, and the this flexibility allows the integration of networking environments. Presently, the internet technology is being run over a wide range of link technologies with varying features in terms of communication rate and bandwidth. The exponential emergence of the internet, increasing need of internet by users, congestion of network traffic and limited bandwidth environment has resulted in the further shrinkage of available limited bandwidth and increase in congestion of network traffic [1].

The main Internet protocol that bears about 90% of the internet traffic in the different wired and wireless networks that are currently in use, is known as TCP [2, 3]. The use of TCP is commonly employed in connection oriented transport layer protocol which is capable of providing dependable packet delivery over links that are unreliable.

The central idea behind the designing of TCP is that wired networks are dependable and any loss of segment that occurs during transmission is as a result of network congestion rather than caused by a medium that is unreliable [4]. This idea is not applicable in the wireless parts of the network. The unreliability of wireless links is high, and segments are always lost because of several factors, which include interruption, higher rate of bit error, fading and mobility-related processes like handovers [5].

The sliding-window protocol specifications are the least requirements for TCP performance. There are many problems associated with the different areas of TCP. Therefore, to enhance the adaptability capability of the protocol in terms of new and bandwidth requirements, the protocol has been modified in different way and then newer algorithms have also been used [6]. Recently, some of them are regarded as TCP requirements. There are two types of these algorithms, and they are; “Slow Start” and “Congestion Avoidance”. Through the use of these algorithms, the performance of the TCP sliding window can be enhanced for the purpose of solving certain problems related to network congestions in networks as well as their nodes. The implementation of these algorithms is done jointly, and they operate as one [7].

Out of the other challenging issues, traffic congestion is one of the main issues that causes congestion in traffic, thereby leading to the demand for a practical solution. Based on this, studying the TCP performance at the individual flows share link becomes crucial [8]. Hence, this paper focuses on analyzing the performance of the TCP within an environment that has different dependable connections with UDP traffic flows.

Different designs of TCP flavors depends on the features of wired networks, since TCP is not dependant on an underlying network layers. Nonetheless, the performance of TCP congestion control algorithms may be low in networks that are naturally heterogeneous. The proposal of the TCP congestion control algorithms was earlier made based on the presumption that packet loss is solely caused by congestion. However, there are other factors that lead to packet loss in wireless networks. Some of those factors include, signal attenuation and fading, multipath disruptions, higher bit error rates caused by weather conditions, obstacles, and wireless end-devices mobility [9].

In order to solve the problem of packet loss caused by congestion and other factors, several TCP techniques and algorithms have been proposed. Classic examples of such techniques and algorithms that have been proposed include TCP Reno, TCP Tahoe, TCP Vegas, TCP New Reno, TCP Reno with Selective Acknowledgement (SACK), and TCP Binary Increase Congestion (BIC). While Snoop-TCP is an approach used in link layer control, M-TCP and I-TCP are approaches of split connection. All of these have been proposed as a means of improving the performance of the network [10,11]. In spite of all the algorithms and techniques that have been proposed, the most promising one is the end-to-end techniques, the only changes required are to the end systems instead of to the intermediate nodes. The use of these end-to-end control approaches is employed in the present day deployed networks.

TCP architecture, which is one of the commonly used transmission layer protocol on the internet consists of four major layers. This architecture allows the end-to-end transmission of non-real-time data. Basically, all internet applications are supported by the TCP/IP architecture [12]. For the lower layers of TCP, there are no defined requirements, because it is assumed that the datagram services provided by the lower layers is unreliable. The upper layers are made up of applications, while the lower layers consist of the IP protocol. The realization of transmission of asynchronous data for the application of TCP protocol is possible [13]. When the network is quiet unreliable, reliable and connection-oriented transmission of data can be achieved by paying more attention to traffic and feasibility control. The aim of this is to provide multiple applications with different interfaces. At the receiving end, buffers that are capable of preserving unread data is received by the TCP. In addition, for TCP to be connection-oriented, it must be able to solve the problems that are associated with a connection. Again, another issue that is of concern in TCP is the safety of communication [14].

In this paper, the three TCP algorithms for congestion control (Slow start and Congestion avoidance (no drop), Fast retransmit (drop no fast), and Fast retransmit and recovery (drop fast)) in wired network are compared using a performance analysis. More so, the use of OPNET Modeler is employed in this study to analyze combined congestion Avoidance and Slow Start algorithms [15]. A survey of congestion controller algorithms is presented in Section 2, while a description of the simulated network topologies is provided in Section 3. Section 4 presents the scenarios of simulation as well as the results for simulation, and Section 5 presents the conclusion.

2. Materials and Methods

Different kinds of congestion control algorithms exist, and they include Retransmission, Flow control end-to-end, time-out and receipt of the duplicate, TCP congestion control, Control of congestion's end to end, Standard TCP congestion algorithms, and Uncontrolled congestion control techniques.

The need for increased internet capacity throughout all network layers is due to the popularity and increased use internet, with home users needing a package that is more than 64 KB/S second, which is often provided by network providers for games. Video and music. The Majority of the problems present in the TCP are associated with used or unused algorithms. The performance of the network largely depends on the effective deployment of TCP, which is widely used on the internet.

2.1. Slow Start and Congestion Avoidance

The control of initial flow of data at the start of a communication and during recovery of error based on the reception of acknowledgements is carried out through Slow Start. The implication of each acknowledgement is that a set of segments or a segment has left the network, and is not making use of any data, thereby allowing new data to be sent.

By means of a Congestion Avoidance algorithm, the transmission of data is adapted to the available resources. The moment a particular threshold is reached by Slow Start, Congestion Avoidance begins to slow down the process of window opening.

The addition of another window is made to the TCP of the sender by Slow start; the congestion window is known as CWND. Upon the establishment of a new connection with a host on a different network, initialization of the congestion to one segment occurs (typically 536 bytes or 512 bytes) [16]. The sender starts by transmitting one segment and waiting for its ACK. Once the sender receives the ACK, an increase occurs in the congestion window from one to two, and it is possible to send two segments. The increase in the congestion window progresses to four when the two segments are acknowledged. This way, exponential growth occurs, even though it is completely exponential because the ACK may be delayed by the sender, typically sending one ACK after receiving every two segments. It is possible for the sender to transmit as low as the leas congestion window as well as the advertised window.

At a certain stage, the internet capacity can be reached, and the elimination of packets from the intermediate router begins. This is an indication to the sender that the congestion window has exceeded its capacity. One of the methods of dealing with packet loss, is Congestion avoidance. The arrival of data on a big pipe (a fast LAN) and the outputs on a smaller pipe (a slower WAN) results in congestion. The arrival of multiple input streams to a router of smaller capacity that is less than the total inputs can lead to congestion [17]. Packet loss is indicative of two things; occurrence of time out or receipt of duplicate ACKs. However, it is generally assumed by the algorithm that the damage caused by the loss of a packet is very minimal (much lesser than 1%). Thus, when packet signals are lost, it means congestion has occurred somewhere in the network between the source and destination.

Despite the fact that congestion avoidance and slow start, operate are different algorithms with various goals their implementation is carried out jointly. The transmission rate of packets to the network must be slowed down by the TCP when there is congestion, and then the slow start is initiated again in order to normalize the process [18].

Without the use of Congestion Avoidance and Slow Start algorithms in TCP, as much information as the window allows will be sent. When this occurs, it may cause greater havoc because this information must be stored by the in the queues of nodes located along the network, and this can make the nodes run out of memory. The inability of a node to store incoming data results in segments loss and occurrence of transmission error. However, when there is no control of flow, the usable window will be filled up immediately, thereby leading to congestion again.

2.2 Reno and Tahoe TCP

The efficiency of the Fast Retransmit and Fast Recovery is higher when it is just one segment that is lost, but when multiple segments drop, problems emerge. It is due to this that the different flavors of TCP exist. The use of these algorithms is employed by some of them, while it is not in some, and the reason for this is to enhance their performance under different conditions. For instance, when just one segment is lost, the performance of TCP can be optimized through the joint implantation of Fast Retransmit and Fast Recovery [19]. Such joint implementation can be of great use when TCP is operating in a network that is reliable with low BER, burst of errors and causing the loss of just one segment. Reno TCP refers to the flavor that employs the use of Fast Recovery and Fast Retransmit.

The occurrence of an error that causes multiple segment loss leads to the emergence of problem with Reno TCP’s congestion window. The congestion window is shut by the Fast Retransmit any time a non-duplicated acknowledgement is received. In an event that multiple segments are lost, the window decreases to half of its normal size for every segment that drops. Moreover, the larger the amount of information sent, more the window that is usable decreases in the fast recovery stage. This occurs each time a duplicated acknowledgement is received. In an event that the congestion window divides into two again, it then shuts down to Zero, thereby barricading the communication and causing forced timeout. Upon the expiration of the retransmission timer, the congestion window closes in 0, and begins increasing with the help of slow start and congestion avoidance algorithms.

2.3 Combined Congestion Avoidance and Slow Start Algorithms
The requirement of the combined congestion avoidance and slow start algorithms is the maintenance of two variables for each connection:

- A congestion window (cwnd).
- A slow start threshold size (ssthresh)

The operation of the combined algorithm is as follows:
The Initialization for a given connection sets cwnd to one segment and ssthresh to 65535 Bytes. It is required the cwnd value be less than or equal to 2*SMSS bytes and must not exceed 2 segments. Sender Maximum Segment Size (SMSS), refers to the maximum size of the segment that the sender is allowed to transmit. The initial value of cwnd may be arbitrarily high, the size of the window which is advertised may be utilized by some implementations, but this may decrease as a reaction to congestion. Usually, TCP output never sends more than the minimum of cwnd and receiver's Advertised window. In a case whereby there is congestion, one-half of the current window size is stored in the ssthresh. In addition, cwnd is set to one, if the congestion is indicated by a timeout. When there is a timeout of when a duplicate ACK is received, it implies that there is congestion. Lastly, the acknowledgement of new data on the other end results in an increase in cwnd. The increase in cwnd is determined by if the TCP is executing a slow start or congestion avoidance task. A cwnd that is less than or equal to SSTHR is indicative of a TCP slow start, and if otherwise, it means that a congestion avoidance operation is being performed by TCP. Slow start progresses till TCP is halfway to where it was before the occurrence of the congestion, and afterwards, congestion avoidance takes over. This occurs as a result of the recorded half size of the window which led to the problem. As earlier stated, the congestion window is exponentially increased by slow start.

On the other hand, a congestion avoidance requires the congestion window (cwnd) to be increased by segsize * segsize / cwnd each time an ACK is received, where segsize is the segment size and cwnd is maintained in bytes. Consequently, a linear growth occurs in cwnd as compared to the exponential growth that occurs in slow start. It is expected that in every round-trip time (irrespective of the number of ACKs are received in that RTT) the cwnd increase should be at most one segment, while in slow start, the increase in cwnd occurs according to the amount of ACKs received within the round-trip duration [20].

3. Slow Start and Congestion Avoidance Simulation
The use of TCP was employed for network setup, instead of the End-to-End transmission protocol. It is made up of a server located in Baghdad, and one client located in Babil province as illustrated in Figure.1 for the entire network, and Figure 2(a) which is made up of a server and Figure 2(b) for eastern network with a mobile client branch. Different mechanisms were used in analyzing the size of congestion window. It is assumed that these networks are perfectly without having any packet loss.

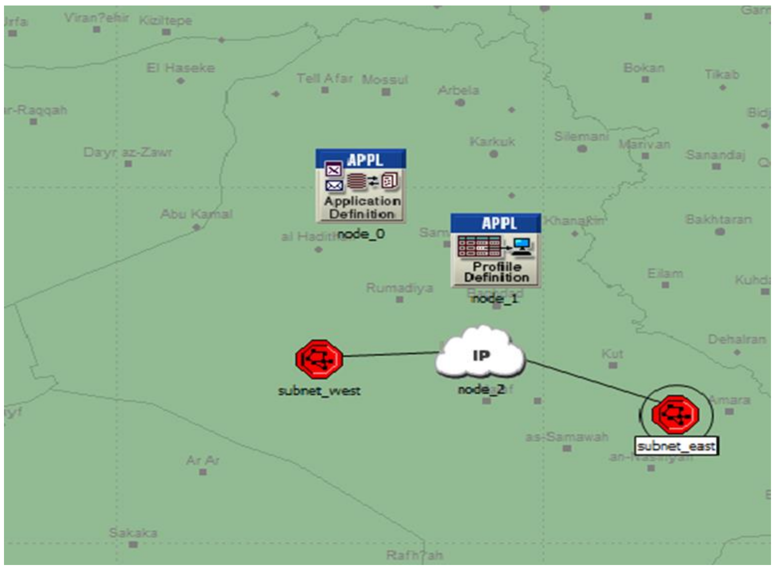

Figure 1. Network figure with OPNET model

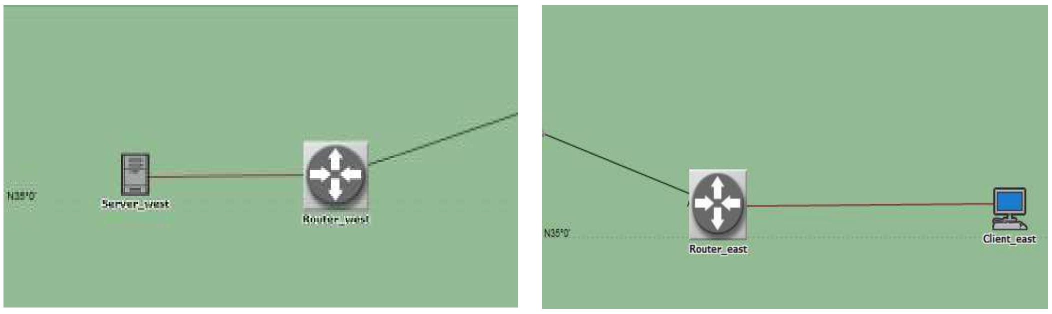

Figure 2. (a)Server and router , (b) client and router located within the FTP server subnet in west and east branch network

3.1 First Scenario: Slow start and Congestion avoidance
For the evaluation of TCP algorithm's performance in wired networks, OPNET model is implemented for a simple client server. In this first scenario, the transport control protocol with “NO DROP” packet is tested. For the implementation of the proposed scenarios in full format, sequence of programmatic programs was implemented.

3.2 Second scenario: Fast retransmit (drop no fast)
Fast retransmit is a variation of the congestion avoidance algorithm. It is expected that the fast transmit algorithm will be used by the TCP for the detection and repair of loss based on incoming ACK duplicates. An indication of the lost segment in the fast retransmit algorithm, is the use of the arrival of 3 duplicate ACKs (4 identical ACKs with no any other intervening packets) arriving. When the 3 duplicate ACKs have been received, a retransmission of what seems to be lost segment is performed by TCP, without waiting for the expiration of the retransmission timer. The initial appearance of the fast transmits algorithm was in the 4.3 BSD Tahoe release for the testing of the Transport Control Protocol with (DROP no fast) of packets.

3.3. Third scenario Fast retransmit & recovery (drop fast)
The execution of Fast recovery Congestion avoidance is performed without slow start after a retransmit sends what seems to be the missing segment. Through this improvement, high throughput is allowed under moderate congestion, particularly for large windows. Slow start is not performed in this case because TCP is notified about the loss of more than one packet through the receipt of duplicate ACK. Since the generation of the duplicate ACK by the receiver is only possible after receiving

another segment, that segment leaves the network and moves to the receiver's buffer. This simply means that data is still flowing between the two ends, and TCP is not interested in reducing the flow suddenly by moving into slow start. The fast recovery algorithm appeared in the 4.3BSD Reno release. Usually, the implementation of fast recovery, and fast retransmit algorithms is carried out jointly as describe below:

a. Upon the receipt of the third duplicate ACK, ssthresh is set to not more one-half of the present congestion window (cwnd), but must be up to two segments and not less than that. The missing segment is retransmitted, with cwnd set to ssthresh, in addition to 3 times the segment size. This way, the congestion window is increased by the number of segments which have moved out of the network and are cached by the other end.
b. With the arrival of another duplicate ACK, an increase occurs in the segment size. If this occurs, the congestion window is inflated for the extra congestion window which has exited the network.
c. The transmission of a segment (packet) is only possible if the new cwnd value and advertised window permits.
d. With the arrival of the next ACK, which implies new data, cwnd is set to ssthresh. This ACK should be the acknowledgment of the retransmission from step a, one round-trip time after the retransmission. Furthermore, every intermediate segment that has been sent between the packet which is lost and the receipt of the first duplicate ACK should be acknowledged by this ACK. This step is congestion avoidance, because TCP is reduced to one-half of its previous rate at the time the packet loss occurred.

4. Simulation Results

Simulations of all the scenarios were performed and the simulation results for the different scenarios are presented subsequently.

4.1. A link of ppp DS1

4.1.1 First scenario (no drop): from the results of the simulation shown in figure 3, it was observed that the increase in the size of the window and the segment sequence number implies that the data has been integrated when no packets are lost and no congestion was detected. This implies that the delivery and receipt of data packets occurred.

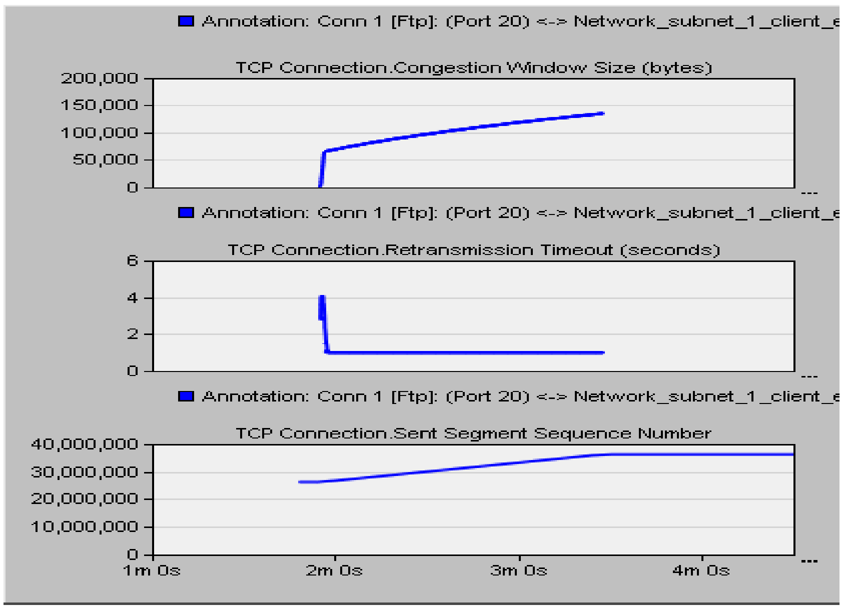

Figure 3. OPNET simulation of the no drop network

4.1.2. A Second scenario (drop no fast): it is important to note that, the steady increase in the window size (bytes) and segment sequence number does not necessarily mean that there is congestion which causes a decrease in the congestion window. This is because 5% of the packets will be abandoned and retransmitted. This is presented on through the horizontal lines of the sequence number graph. It can also be observed that any time a packet is abandoned on the congestion window, the segment sequence number increases, as explained in figure.4.

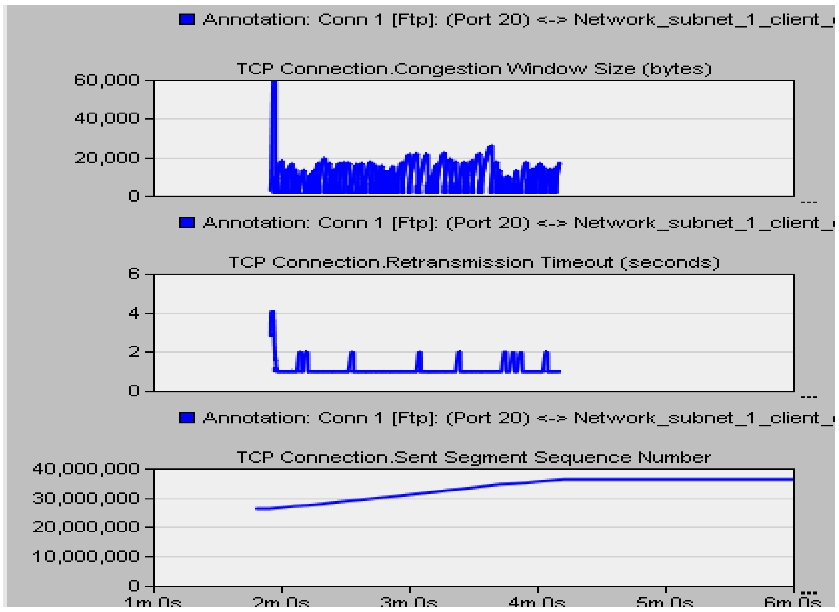

Figure 4. OPNET simulation of the drop no fast network

4.1.3. Third Scenario (drop fast):

The congestion window seems to be quiet unstable because of because of the rapid transformation of the congestion window. From the graph of the sequence number of the frame, very minimal aliases caused by small horizontal lines were observed. This appearance can be attributed to the fact that very minimal values were contained in the packets, and the reason for these small areas return to the use of the fast retransmit algorithm.

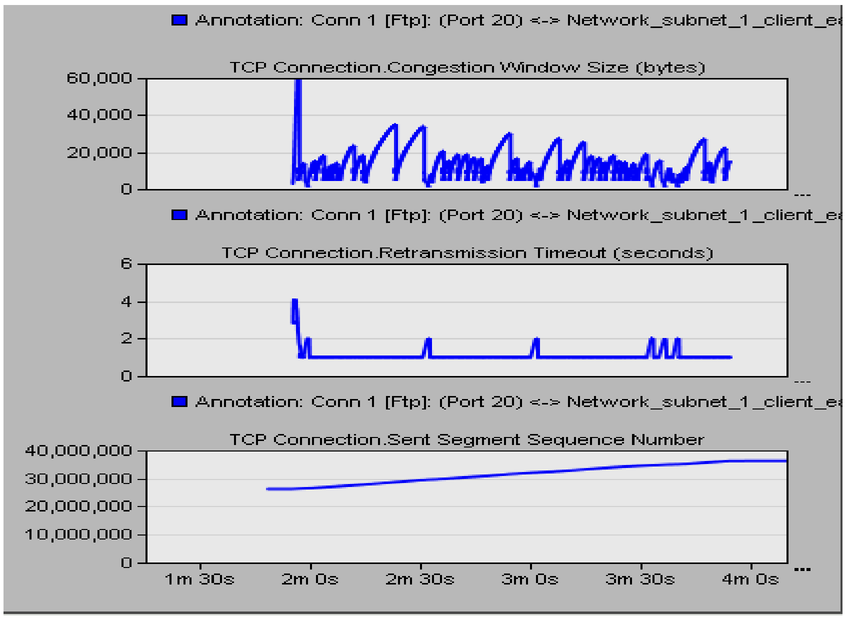

Figure 5.OPNET simulation of the drop fast network

4.2. A ppp DS3 link between east & west subnets, and ip32 cloud internet

4.2.1. First Scenario (no drop): based on the simulation results of the second scenario, the increase in the window size (bytes) and the segment sequence number greater than the first scenario, implies that data has been integrated, with no record of packet loss of congestion. This simply means that the delivery and receipt of data occurred respectively.

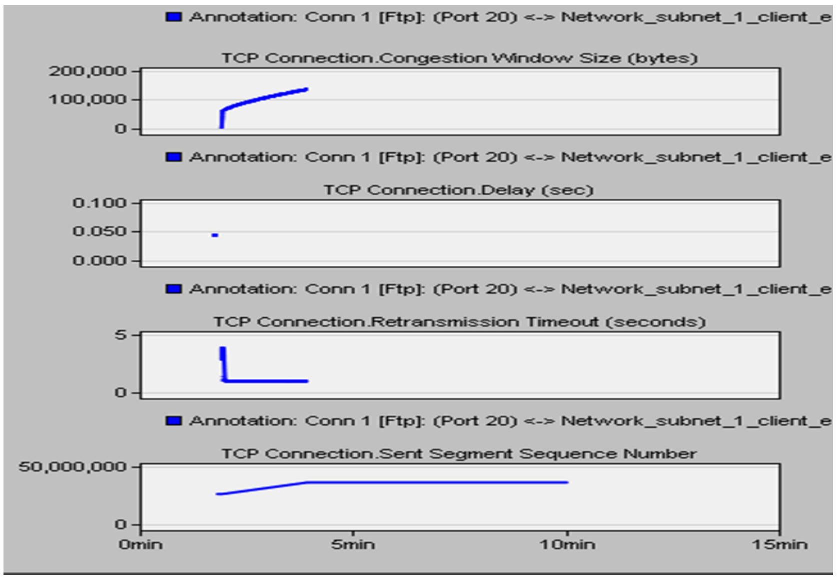

Figure 6. NO drop simulation network

4.2.2. Second scenario (drop no fast):

It was also observed that the increase in both the window size (bytes) and segment sequence number is not more than that of the first scenario as shown in figure. 7, with reference to the fact that, when congestion occurs, there is a decrease in the congestion window. This caused by the fact that 5% of the packet will be abandoned and retransmitted. This presented on the horizontal lines of the sequence number graph curve. It was also observed that anytime a packet is neglected on the congestion window, the segment sequence number increases.

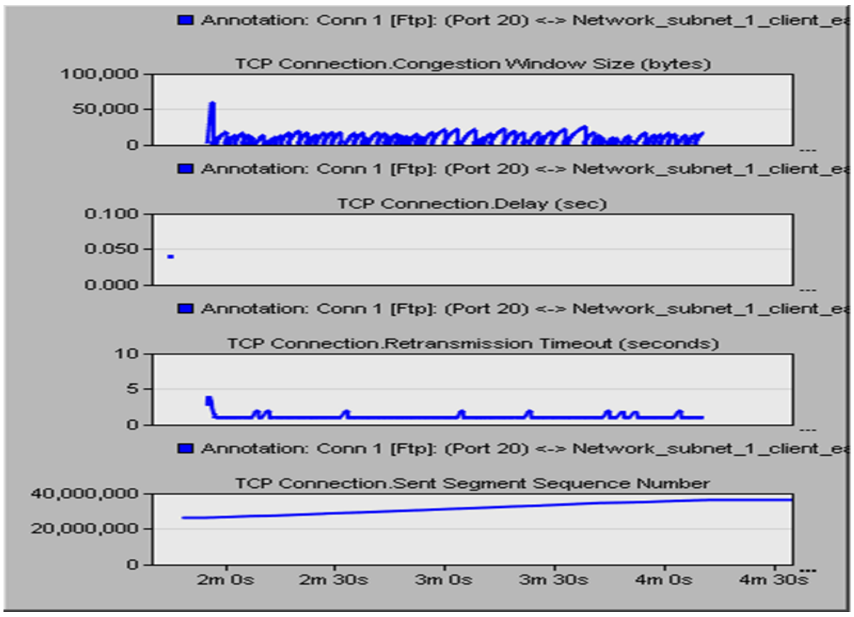

Figure 7. drop NO fast simulation

4.2.3. Third Scenario (drop fast)

The congestion window seems to be somehow unstable with a steady increase in the serial number of the frame, and this increase is significantly greater than that of the first scenario because of the rapid transformation in the state of the congestion window. By checking the graph of the sequence number of the frame, it was observed that there are minimal aliases caused by small horizontal lines. They are present because the packet was full of very small values.

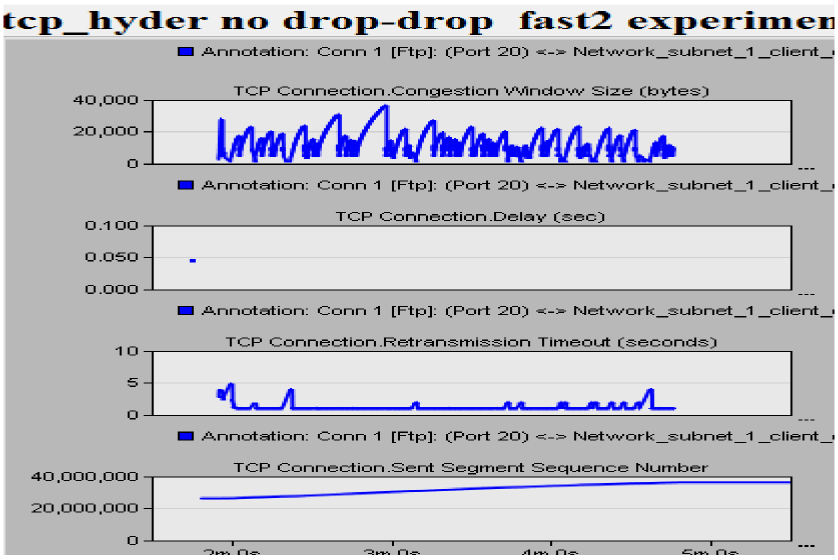

Figure 8. drop fast simulation

4.3 Comparison between both scenarios

The comparison between the two scenarios, first with congestion control algorithms when the link is ppp DS1 and a second scenario of congestion control algorithms when the link is ppp DS3 is described as follows:

4.3.1. Case 1: no drop (congestion window & sent sequence number)

As shown in figure. 9, the no drop scenario when the link between routers and ip32 cloud internet is a PPP DS1 (blue), and when the link between routers and ip32 cloud internet is a PPP DS3 (red), a very small difference is shown.

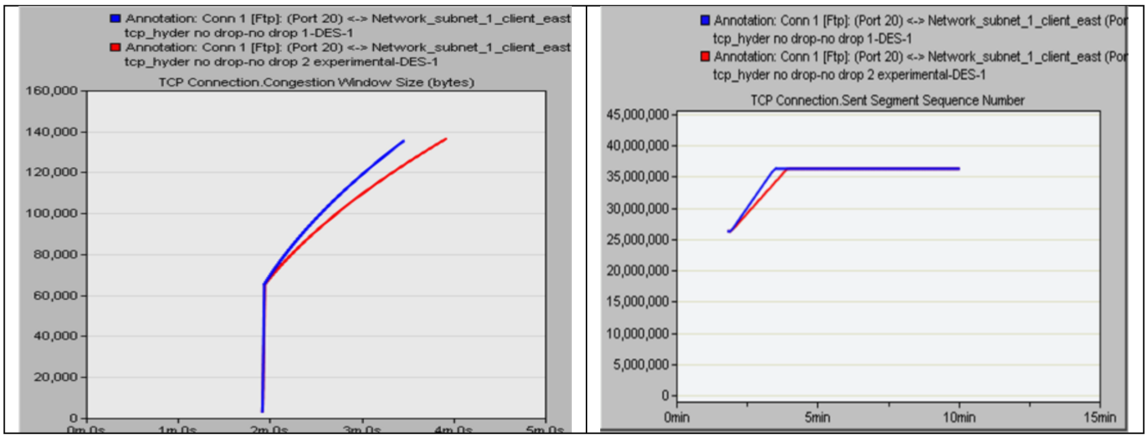

Figure 9. no drop scenario comparison

4.3.2. Case 2: drop no fast (congestion window & sent sequence number)

As shown in figure. 10, there is a convergence between the two links as expected.

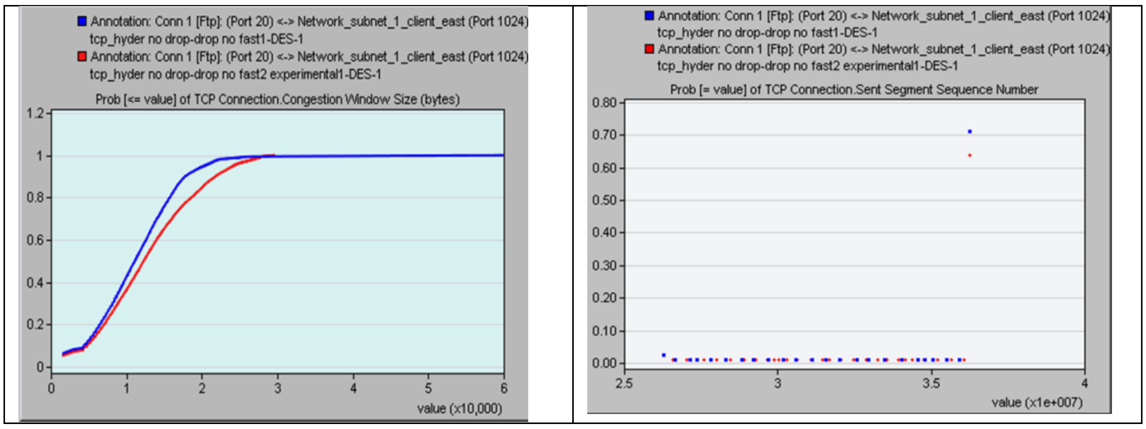

Figure10. a comparison result of drop no fast

4.3.3. Case 3: drop fast congestion window & sent sequence number

In drop fast comparison shown in figure. 11, a PPP DS3 link, red curve , have a peak horizontal value more than a PPP DS1 link due to the application of fast congestion.

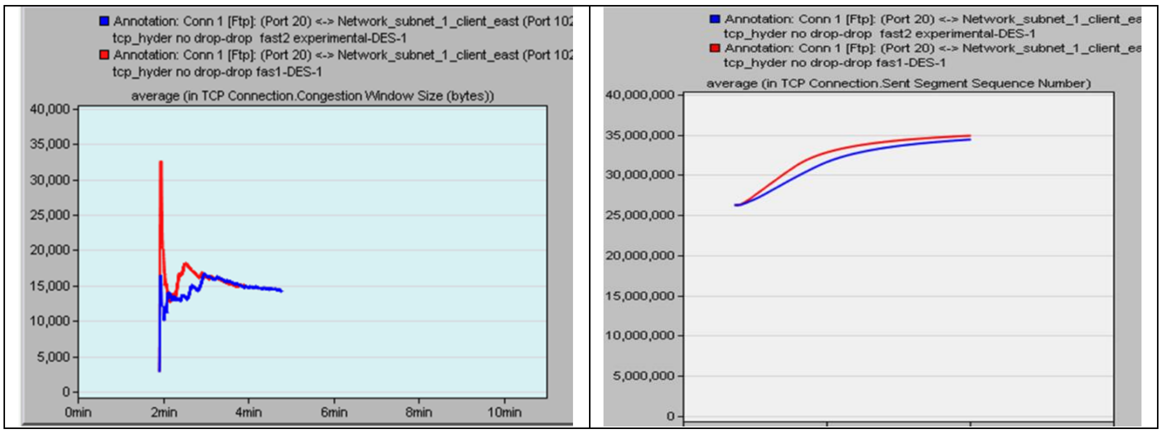

Figure11. drop fast comparison between PPP DS1 and PPP DS3 link

5. Conclusion:

- The first scenario which is a “No Drop” scenario presented in the first graph, shows that there was no packet, but as any changes occur in the link between ppp DS1 to ppp DS3, the sent segment sequence number and congestion window increase.
- The second graph is an illustration of the second scenario which is the Tahoe scenario. In this scenario, a packet loss of 0.5% was recorded. The indication of congestion through timeout, resulted in setting cwnd to one segment. This simply implies that slow start which increase in if the link was ppp DS3, was executed.
- The third graph is an illustration of the Reno scenario in which packet loss of 0.5% was recorded. Based on the observation of the Tahoe graph, the congestion window size does not drop to zero. Rather than performing slow start which is reduced in the case of ppp DS3, fast recovery was performed.

References

[1] Soma M, Arun K M, Pratiti R 2016 *Enhancement of TCP Performance over MANET International Research Journal of Engineering and Technology* **3** Issue: 05.

[2] Yakubu F, Abdullahi S E and Aigbe P E 2011 *Performance Analysis of TCP in a Reliable Connection Environment with UDP Flows using OPNET Simulator International Journal of Scientific & Engineering Research* **2** Issue 12.

[3] Fitigau I, Toderean G 2012 Performance analysis of TCP and UDP using Opnet Simulator 11th *International Conference on Development and Application Systems*, Suceava, Romania, May 17-19.

[4] Mostfa M K 2010 TCP versus UDP performance in term of bandwidth usage University Utara Malaysia.

[5] Elsadig G E Karar 2018 TCP Flavors Simulation Evaluations over Noisy Environment *International Journal of Information Engineering and Applications* **1** 11-17.

[6] Gital A, Ismail A S, Chiroma H and Abdullahi M S 2014 Packet Drop Rate and Round Trip Time Analysis of TCP Congestion Control Algorithm in a Cloud Based Collaborative Virtual Environment *UKSim-AMSS 8th European Modelling Symposium*.

[7] Yew B, Ong B and Ahmad R 2011 Performance Evaluation of TCP Vegas versus Different TCP Variants in Homogeneous and Heterogeneous Networks by Using Network Simulator 2 *International Journal of Electrical & Computer Sciences* **11**.

[8] Gital A Y and Ismail A S 2014 A Framework for the Design of Cloud Based Collaborative Virtual Environment Architecture i*n Proceedings of the International MultiConference of Engineers and Computer Scientists*, pp. 196-200.

[9] Subedi L, Najiminaini M and Trajkovi L 2008 Performance Evaluation of TCP Tahoe, Reno, Reno with SACK, and NewReno Using OPNET Modeler Communication Networks Laboratory http://www. ensc. sfu. ca/research/c nl OPNET technologies.

[10] Hussein A M, Adnan H A 2013 Effect of some security mechanisms on the Qos VoIP application using OPNET *International Journal of Current Engineering and Technology* **3** Issue 5.

[11] Ikram U D, Mahfooz S, Rahat ullah and Zeeshan M 2013 Demonstration of the Congestion Control Algorithms Implemented by TCP *World Applied Sciences Journal* **22** 877-881.

[12] Adnan H A 2015 Performance Evaluation of Wi-Fi Physical Layer Based QoS Systems on Fiber Using OPNET Modeler *International Journal of Soft Computing and Engineering (IJSCE)* **5** Issue-3.

[13] Aggarwal A, Stefan S, Thomas A 2000 Understanding the Performance of TCP Pacing, March 30, 2012, IEEE InfoCom.

[14] Sally F, Kevin F 1999 Promoting the Use of End-to-End Congestion Control in Internet", IEEE/ACM Transactions on Networking, August.

[15] Adnan H A, Ali N A, Maan H H 2013 Performance Evaluation ofIEEE802.11g WLANsUsing OPNET Modeler *American Journal of Engineering Research (AJER)* **2** 09-15.

[16] Van J 1990 Modified TCP Congestion Control Avoidance Algorithm. end-2end-interest mailing list, April 30.

[17] Sally F 1994 TCP and Explicit Congestion Notification *ACM Computer Communications Review* 10-23.

[18] Adnan H A and Ahmed R A 2016 OPNET Scenarios of WiFi and WIMAX Networks Performance Analysis *International Journal of Innovative Technology and Exploring Engineering* **6** Issue-4.

[19] Floyd S, Henderson T 1999 The New Reno Modification to TCP’s Fast Recovery Mechanism, April, RFC 2582.

[20] Harris Interactive P C 2010 Internet Use Continue to Grow at Record Pace. Press Release, February 7.