Analysis of Network Traffic Congestion Control over TCP Protocol

Muhammad Kashif Hanif, Sheikh Muhammad Aamir, Ramzan Talib, and Yahya Saeed,

Government College University, Faisalabad, Pakistan.

Summary
The network performance is affected due to congestion. The congestion is occurred due to overloading data over low capacity handling node. The objective of this research is to investigate the applications (like email, file transferring/downloading, web browsing and remote login) performance in the perspective of application reply time, throughput, queuing delay, Transmission Control Protocol (TCP) Delay over Point-To-Point Protocol E1 with TCP default and variable window sizes (8K, 32K, and 64K). This study also dealt with ATM and Frame Relay E1 links with Nagle's Algorithm for small data packets with large size TCP/IP headers delivery over the internet. The extent of this research is incorporated scenario based investigation of various protocols utilized at end-to-end and intermediary devices using OPNET (Optimized Network Engineering Tool) network simulator.

Key words:
Point-to-Point Protocol, Frame Relay, Asynchronous Transfer Mode, Queue Delay, Transmission Control Protocol

1. Introduction

As the public network i.e. internet is growing rapidly, various kinds of network traffic which includes textual data, video, audio streaming etc. are flowing over the network. As the web is a diverse network in which moderate speed systems are additionally converging in fast systems. Distinctive applications which utilized conventions like FTP, Telnet, Email, WWW, and HTTP and so forth are running on the web and system clog is happened when a few parcels are sent to the system over its ability or as it were system assets move toward becoming over-burden [1]. Due to congestion over the internet affects the network performance with respect to packet loss, delay in packet receiving and Throughput etc. Network performance diminishes due to expansion in packet reduction and intermediary devices queue delay, which causes the throughput degradation.

For administrating computer network resources, a congestion control algorithm shows a significant role in network data flow engineering. Various types of algorithms (like Fast Recovery, Fast Re-transmission, Congestion Avoidance, Slow Start, and adjustment in Transmission Control Protocol like Reno, Vegas, Tahoe, CUBIC TCP, FAST TCP, and Nagle's method and so on.) are utilized for controlling or avoiding the congestion through some fundamental TCP mechanism as TCP is a reliable protocol. A few mechanisms are dealing with packet loss, some are dealing with packet delay, and some have consolidated components.

1.1 TCP Windows Concepts

TCP sliding window algorithm assumes vital part in the utilization of transfer speed and stream control that window size is regularly in bytes. The sliding window is used in transport layer for flow control that is a variable indicated in TCP segment header by the receiver device as per its windows capacity that directs the sender device to send a particular number of bytes [2], while congestion window is kept up at sender side which is one element that decides the quantity of extraordinary bytes whenever. The window extension can be ascertained by evaluating how much blockage between two points exists. When the link is being set up the congestion window is adjusted by maximum segment size (MSS).

1.2 General Principles of Congestion Control

The fundamental subject of congestion control systems is to maintain the data flow underneath the limit of network devices generally in term of transmission capacity over the network. There are following two sorts of congestion control mechanisms:

1) Open loop congestion control mechanisms i.e. Avoidance Policies
2) Close loop congestion control mechanisms i.e. Expulsion Policies

In first type of control, congestion control is maintained at sender or at target but in close-loop, express (switch or switches enlighten the source concerning the congestion event) and verifiable (source may settle on delay premise i.e. network is congested or not) inputs are utilized.

Manuscript received July 5, 2017
Manuscript revised July 20, 2017

1.3 TCP Handling of Network Congestion

Transmission Control Protocol is a process-to-process or an end-to-end correspondence and it underpins open-loop congestion policies that mean prevention. In Transmission Control Protocol, congestion can be maintained or evaded with its fundamental congestion control policies i.e. Fast-Recovery, Fast-Retransmission, Congestion Avoidance, and Slow Start. In this method, information loss or communication delay might be detected. Network congestion over the TCP is handled by using AIMD [3][4][5], Transmission Control Protocol Fast-Recovery, Fast Retransmission, Congestion Avoidance, and Slow Start [6][7]. Different standards and advanced TCP algorithms are TCP Reno, New Reno, Selective Acknowledgement – SACK High Speed TCP – STCP, Scalable TCP – STCP, TCP Vegas [4] [7][8].

1.4 Nagle's Algorithm

Now and then the computer network congestion is happened because of sending of numerous little sizes data packets over the computer network. As the size of Transmission Control Protocol header is forty (40) bytes in which 20 bytes are of IP and if the sending data size is forty one (41) bytes, there is only a single byte that is valuable. Along these lines, these sorts of extensive number of data packets over the computer network take the system to blockage crush. This sort of issue is for the mostly happened in the TELNET applications. To tackle this issue John Nagle built up a method named as Nagle's Algorithm [9]. This mechanism is utilized to enhance the proficiency of TCP/IP arrange by consolidating these little data packets and after that send it as a whole over the network. This procedure of connecting is known as Nagling. Along these lines, the Nagle's mechanism trains the source device to store information in their buffer until that buffer is full or new acknowledgement is gotten [10]. The calculation is given underneath:

```
Nagle's Algorithm (MMS, Window size)
If new data to send, then:
   If window size > = MSS & available data > = MSS then:
      Send complete MSS segment
   Else
      If there is outstanding data in pipe, then:
Buffer new data until an ACK is received
      Else
Immediately send data
       [End of if structure]
    [End of if structure]
[End of if structure]
```

Nagle's algorithm collaborates most noticeably bad with TCP packet delay acknowledgement. The applications which are real time, for example, online multi-player recreations, telnet, and so forth require quick reaction for any activity in the diversion. Thus, there is no requirement for Nagle's algorithm in such kind of applications. TCP_NODELAY socket choice is utilized for this reason.

2. Review of Literature

Akujobi et al. proposed a modern algorithm based on combined features of Backward Explicit Congestion Notification (BECN) and Explicit Congestion Notification (ECN) in TCP/IP computer networks. A comparative study is made and it is assessed that the ECN+BECN instrument fundamentally decreases line vacillations because of early congestion sign contrasted with ECN. It is likewise concentrated that ECN+BECN plan can fundamentally lessen the Source Quench of Internet Control Message Protocol - ICMP switch activity in a network contrasted with a BECN network [4].

Talaat et al, examined two sorts of network congestion control mechanisms which are regular network congestion control calculations and media (varying media substance) congestion control mechanisms. In this research study they demonstrated that substance mindful over web prompt better network congestion control plans, it is just because of bigger media activity over the web [5].

Onwutalobi, and Claret described that the TCP has some basic mechanisms known as Fast Retransmit, Fast Recovery, Congestion Avoidance and Slow Start by utilizing these mechanisms, the network steadiness is obtained proficiently provided that extraordinary connection application and minimum queuing suspension [6].

Mathis proposed another mechanism with the straightforward alteration of AIMD which was Relentless Congestion control. By the use of this method congestion window is decreased by the quantity of lost sections as opposed to dividing cwnd window. Relentless congestion control strategy makes arrange devices less demanding to precisely control the network traffic [10].

Sangtae et al., they looked into on the effect of background network transportation on the efficiency of rapid Transmission Control Protocol mechanism including H-TCP, CUBIC, HSTCP, FAST, BIC-TCP, and Scalable Transmission Control Protocol. They show that the join use, speed union, solidness and reasonableness of the conventions are influenced by the distinction in stream sizes, Round Trip Time, and measure of foundation streams contending with fast streams in an overfilled router. They have bring into being that the round trip times RTTs and existence of background network traffic enhances the fairness of numerous rapid mechanisms, all high velocity protocols/mechanisms barring FAST and High Speed

Transmission Control Protocol demonstrate great intra protocol fairness paying little respect to a background activity. High Speed TCP needs a lot of background traffic, H-TCP bargains steadiness of fairness, and in little and extended delayed computer networks, FAST mechanism is influenced by wrong and variability rather than background activity [11].

Parsa, and Garcia-Luna-Aceves, they depicted another mechanism for network congestion control in Transmission Control Protocol, i.e. TCP Santa Cruz which depends on utilizing assessments of packet delay beside the forward way, as opposed to RTT delay and creating versatile utilization of any acknowledgements received over the window instead of increment in congestion window. The Transmission Control Protocol Santa Cruz additionally demonstrated the course of network congestion and separates the onward throughput from events on the turnaround path [12].

Srinivas et al. dealt with congestion Control mechanism in Transmission Contorl Protocol i.e. TCP. In this research they presented the Transmission Control Protocol congestion window for the source being autonomous of data packet misfortunes, and maintaining the transmission at an indistinguishable rate from some time [13]

Jamal, and Sultan exhibited that Transmission Control Protocol has unique and various flavors for congestion controlling in the network traffic flow. Some are based on packet delay, while some packet loss and some have regular elements of both cases as discussed before, as far as congestion window versus passed time after the connection has been made [14].

Chandrayana, Sikdar, and Kalyanaraman performed experiment on performance analysis between Additive Increase and Multiplicative Decrease (AIMD) and Binomial Congestion Control Scheme (BCCS) that was newly proposed mechanism for congestion control in TCP. This comparative analysis depended on throughput, timeouts, packet loss, fairness and self-similitude. AIMD diminishes the data damages and delay which builds the throughput and reduces the level of self-similarity of the network traffic movement. It is additionally noticed that with extensive adequate number of streams, Binomial Congestion Control Scheme contend reasonably with Transmission Control Protocol [16].

3. Methodology

For the purpose of this research, a scenario based study is performed using a simulator OPNET 14.0 which is used to analyze the performance and activities of computer network regarding various applications.

A scenario is developed for this research in OPNET 14.0 simulator which is described in the following section

3.1 Scenario in OPNET

For this study, network is framed on the Pakistan (world map), Karachi as Head Office and one Regional Office at Islamabad by utilizing OPNET project editor. There are add up to eighteen (18) devices utilized for the whole network scenario which incorporates two router (CISCO 7507, 2514), two network switches, ten PCs and four servers machines. The connections are used like Serial connection E1, Ethernet, ATM with E1, and Frame Relay with E1 joins.

A Data Center is developed in the Head office, with four application servers including File Transfer Protocol, Email, Terminal servers and Web server. These servers are associated with core network switch and this switch is again associated with Core Router.

The Islamabad office has ten (10) PCs which are associated with Access switch. The Head Office-Karachi and Regional Office-Islamabad are connected with layer 2 protocols like Point-to-Point Protocol with E1, ATM with E1, and Frame Relay with E1 to investigate the efficiency of computer networks as indicated by various situations in regards to congestion control in these advancements.

There are eight (8) different cases used for the above said scenario used to analyze the Response Time regarding (HTTP, Email, File Transfer Protocol, and Remote Login) as per various layer2 protocols (Point-To-Point Protocol, ATM, and Frame Relay) with E1 link in the simulator. The detail of layer 2 protocols with E1 link and TCP window sizes is as follows:

1) Point-To-Point Protocol E1 link and default TCP window size
2) Point-To-Point Protocol E1 link and 8K TCP window size
3) Point-To-Point Protocol with E1 link and 32K TCP window size
4) Point-To-Point Protocol with E1 link and 64K TCP window size
5) ATM with E1 link and 64K window sizes
6) Frame Relay with E1 link and 64K window sizes
7) ATM with E1 link 8K window size with Nagle Algorithm
8) ATM with E1 link 64K with Nagle Algorithm

4. Results

The comparison is made to check the performance of various applications (File Transfer Protocol, Email, Hyper Text Transfer Protocol, Remote Login and TCP Delay response times) using Point-To-Point Protocol, ATM and Frame Relay E1 link with TCP default, 8K, 32K and 64K window sizes and also comparison is performed with Nagle's Algorithm.

In section 4.1, the comparison is performed for different applications using Point-To-Point Protocol E1 link with different TCP windows sizes. In section 4.2, the comparison is performed for different applications using Asynchronous Transfer Mode (ATM), Point-to-Point Protocol, and Frame Relay E1 with the window sizes 64K. In section 4.3, the comparison is performed for different applications using ATM with E1 and 64K TCP window size, ATM with E1, with 8K TCP window size Nagle and ATM with E1, 64K windows size having Nagle method.

4.1 Point-to-Point Protocol with E1 link and default TCP Window size vs. 8K, 32K, and 64K TCP windows sizes

4.1.1 Email Response Time

The graphs in the results sections are represented as x-axis coordinate represents time (minutes) and y coordinate represents response time.
According to simulation outcomes about for Email reaction time, Point-to-Point Protocol E1 with default TCP window measure showed stable conduct when contrasted with others if there should arise an occurrence of Email download reaction. Along these lines, Point-to-Point Protocol with E1 link and default TCP window measure versus 8K, 32K and 64K demonstrates better reaction time for emails downloading is appeared in Fig 1.

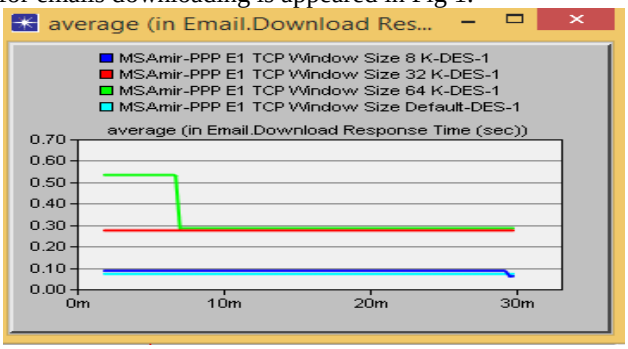

Fig 1. Email Download Response time (Point-To-Point Protocol with E1 link and default, 8K, 32K and 64K TCP window sizes)

The summary of this scenario for Email download response times is shown in Table 1.

Table 1: Email Response Time Point-To-Point Protocol with E1 link and default, 8K, 32K, 64K window sizes

Protocol /Link	TCP Window Size	Email Download Response Time (sec)		
		Avg.	Max.	Min.
Point-To-Point Protocol/ E1	Default	0.0712	0.0712	0.0712
	8K	0.0618	0.0860	0.0357
	32K	0.2744	0.2744	0.2744
	64K	0.2846	0.5335	0.0357

4.1.2 FTP Response Time

Point-to-Point Protocol with E1 link and 64K window size has shown in the Fig. 2, better response in File Transfer Protocol download response time case when contrast with window sizes 8K, 32K and default of TCP.

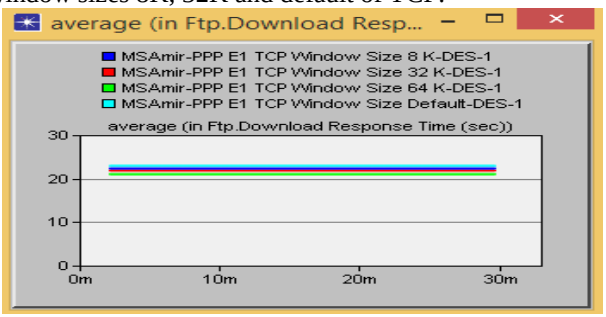

Fig. 2. FTP Download Response Time (Point-To-Point Protocol with E1 link and default, 8K, 32K and 64K TCP window sizes)

The average summary values of Point-to-Point Protocol with E1 link and 8K, 32K, 64K and default TCP window size as per File Transfer Protocol (FTP), download response time as follows in Table 2.

Table 2: FTP Response Time, Point-To-Point Protocol with E1 (default, 8K, 32K, 64K window sizes)

Protocol/Link	TCP Window size	FTP Download Response Time (sec)		
		Avg.	Max.	Min.
Point-To-Point Protocol/E1	Default	22.942	22.942	22.942
	8K	22.319	22.319	22.319
	32K	21.858	21.858	21.858
	64K	21.016	21.016	21.016

4.1.3 HTTP Response Time

Point-to-Point Protocol with E1 link and TCP Default window size has represented far better results in Fig. 3 when contrast with other in web page response time.

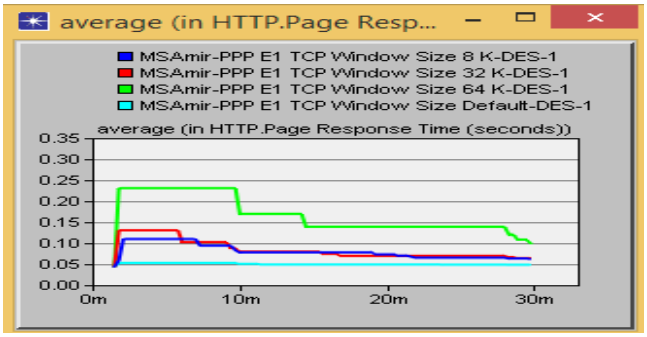

Fig. 3. HTTP Page Response Point-To-Point Protocol with E1 link and default, 8k, 32k, 54k window sizes

The average summary values of Point-to-Point Protocol with E1 link and 8K, 32K 64K and default TCP window size regarding HTTP download response time is shown in Table 3.

Table 3: HTTP Page Response Time, Point-To-Point Protocol E1 (default, 8K, 32K, 64K window sizes)

Protocol/Link	TCP Window Size	HTTP Page Response Time (sec)		
		Avg.	Max.	Min.
Point-To-Point Protocol/E1	Default	0.0486	0.0611	0.0445
	8K	0.0622	0.2091	0.0447
	32K	0.0644	0.2165	0.0424
	64K	0.0995	0.4172	0.0450

4.1.4 Remote Login Response Time

Simulation results showed in Fig. 4 that in case of Remote login reaction time Point-To-Point Protocol with E1 link and TCP default window estimate has preferred reaction over others Point-To-Point Protocol E1 with TCP diverse sizes. The outline estimations of Point-To-Point Protocol with E1 and 8K, 32K 64K and default TCP window measure in regards to Remote Login reaction time is appeared in Table 4.

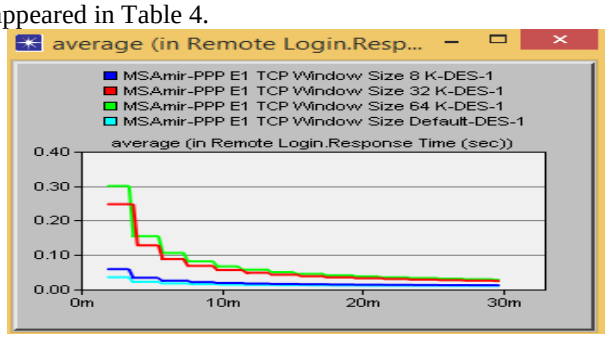

Fig. 4. Remote Login Response time (Point-To-Point Protocol with E1 link and default, 8K, 32K, & 64K window sizes)

Table 4: Remote Login Point-To-Point Protocol with E1 link and default, 8K, 32K, 64K window sizes

Protocol/ Link	TCP Window Size	Remote Login Response Time (sec)		
		Avg.	Max.	Min.
Point-To-Point Protocol/E1	Default	0.0107	0.0356	0.0087
	8K	0.0122	0.0591	0.0087
	32K	0.0247	0.2475	0.0087
	64K	0.0282	0.3005	0.0088

4.1.5 Transmission Control Protocol Delay

Point-to-Point Protocol with E1 link and 8K TCP window size has represented in the Fig. 5, less Transmission Control Protocol Delay than others Point-to-Point Protocol with E1 and TCP window sizes.

The average summary values as per Point-to-Point Protocol with E1 link and default, 8K, 32K and 64K for TCP Delay is as given in the Table 5.

Table 5: TCP Delay Point-To-Point Protocol with E1 link (default, 8K, 32K, 64K window sizes)

Protocol/ Links	TCP Window size	TCP Delay Time (sec)		
		Avg.	Max.	Min.
Point-To-Point Protocol/ E1	Default	1.020	16.126	0.004
	8K	0.817	15.378	0.004
	32K	0.834	15.681	0.004
	64K	0.830	15.363	0.004

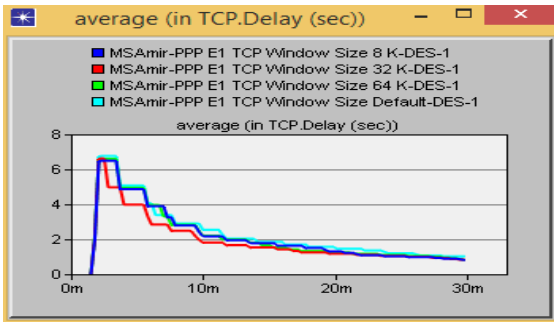

Fig. 5. TCP Delay (Point-To-Point Protocol with E1 link and default, 8K, 32K, & 64K window sizes)

4.2 Point-To-Point Protocol, Frame Relay, and ATM with E1 link and 64K TCP window size

Now the comparison will take place on the behalf of Point-To-Point Protocol, ATM and Frame Relay using E1 link with 64K TCP window size for all applications (Email, HTTP page download, FTP file download, Remote Login and TCP Delay).

4.2.1 Email Response Time

After running the simulation for Point-To-Point Protocol with E1 link and different TCP window sizes, we study the behavior of network topology with Point-to-Point Protocol with E1 link, Frame Relay, and ATM with 64K window sizes.

OPNET results showed that Point-To-Point Protocol with E1 link and 64K TCP window size preferred response time in Fig. 6 than other technologies (ATM and Frame Relay) in the case of Email download response time in the research scenario.

The summary of values regarding Point-to-Point Protocol, Asynchronous Transfer Mode and Frame Relay (all with E1 links) and 64K TCP window size for Email response time is shown in Table 6.

Table 6: Email Download Response Time (Point-to-Point Protocol, Asynchronous Transfer Mode, Frame Relay with E1link and 64K window size)

Protocol with E1 Link and 64K TCP window size	Email download Response Time (sec)		
	Avg.	Max.	Min.
Point-To-Point Protocol	0.2846	0.5335	0.0357
ATM	0.5100	1.4575	0.0363
Frame Relay	0.4130	0.7902	0.0357

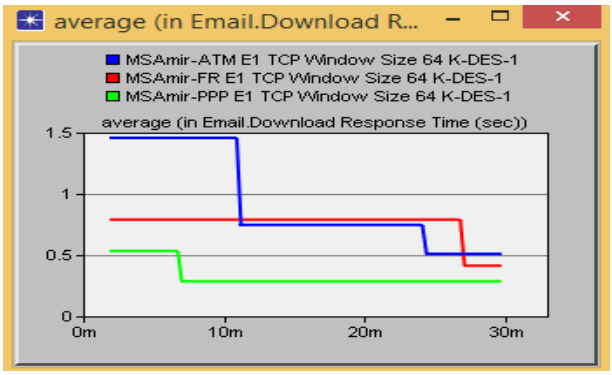

Fig. 6. Email Download Response time (Point-to-Point Protocol, Asynchronous Transfer Mode & Frame Relay with E1 link and 64K window sizes)

4.2.2 File Transfer Protocol Response Time

Asynchronous Transfer Mode with E1 link and 64K TCP window size has represented better response in Fig. 7 than other technologies regarding FTP download response time. The average summary regarding Point-To-Point Protocol with E1, ATM with E1 and FR with E1 and 64K TCP window size for FTP response time is shown in Table 7.

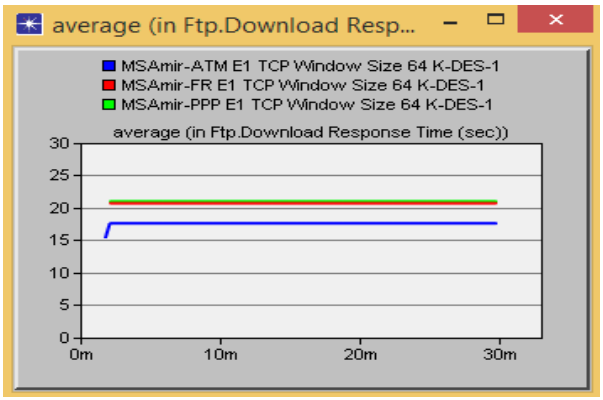

Fig. 7. FTP Download Response time (Point-To-Point Protocol, ATM& FR with E1 and 64K window sizes)

Table 7: File Transfer Protocol Download Response Time (Point-To-Point Protocol, ATM, Frame Relay with E1 and 64K window size)

Protocol with E1 Link and 64K TCP window size	FTP download Response Time (sec)		
	Avg.	Max.	Min.
Point-To-Point Protocol	21.016	21.016	21.016
ATM	17.625	19.751	15.498
Frame Relay	20.724	20.724	20.724

4.2.3 HTTP Page Download Response Time

The investigation of Point-To-Point Protocol E1 64K TCP window measure versus ATM and Frame Relay 64K TCP window measure with respect to HTTP page reaction time has been made. FR E1 with 64K TCP window size Point-To-Point Protocol with E1 link have indicated preferable reaction time over different advancements in regards to HTTP page reaction time. (Fig. 8)

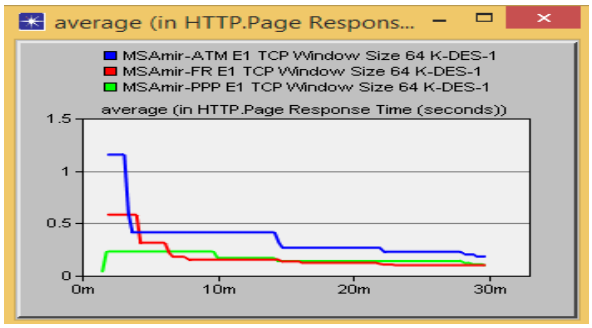

Fig. 8. HTTP Page Response time (Point-To-Point Protocol, ATM& FR E1 with 64K window sizes)

The average summary regarding ATM, Point-To-Point Protocol, and Frame Relay (all with E1 link) and 64K TCP window size for HTTP response time is shown in Table 8.

Table 8: HTTP Page Response Time (ATM, Point-To-Point Protocol, Frame Relay with E1link and 64K window size)

Protocol with E1 Link and 64K TCP window size	HTTP page Response Time(sec)		
	Avg.	Max.	Min.
Point-To-Point Protocol	0.0995	0.4172	0.0450
ATM	0.1827	1.1545	0.0406
Frame Relay	0.1003	0.5822	0.0458

4.2.4 Remote Login Response Time

ATM with E1 link and 64K TCP window size has shown better response (in the Fig. 9) than other technologies (ATM and Frame Relay) regarding Remote Login Response time.

The average summary of Point-To-Point Protocol, ATM and FRAME RELAY (all with E1 links) and 64K TCP window size for Remote Login response time is shown in Table 9.

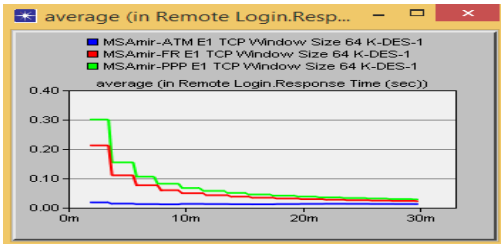

Fig. 9. Remote Login Response time (Point-To-Point Protocol, ATM& FRAME RELAY with E1 link and 64K window sizes)

Table 9: Remote Login Response Time (Point-To-Point Protocol, ATM, and FRAME RELAY with E1 link and 64K window size)

Protocol with E1 Link and 64K TCP window size	Remote Login Response Time(sec)		
	Avg.	Max.	Min.
Point-To-Point Protocol	0.0995	0.4172	0.0450
ATM	0.0124	0.0185	0.0094
Frame Relay	0.0224	0.2123	0.0088

4.2.5 Transmission Control Protocol Delay

ATM with E1 and 64K TCP window size has shown (Fig. 10) quite better i.e. less TCP Delay as compared to ATM and Frame Relay in case of TCP Delay.

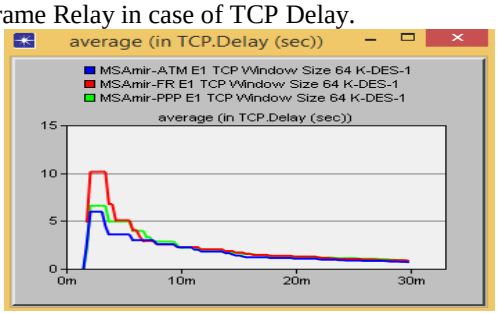

Fig. 10. TCP Delay (Point-To-Point Protocol, ATM& FRAME RELAY E1 with 64K window sizes)

The average summary regarding Point-To-Point Protocol, ATM, and FRAME RELAY (all with E1 links) and 64K TCP window size for TCP Delay response time is also shown in the Table 10.

Table 10: TCP Delay (Point-To-Point Protocol, ATM, FRAME RELAY with E1 link and 64K window size)

Protocol with E1 Link and 64K TCP window size	TCP Delay Time (sec)		
	Avg.	Max.	Min.
Point-To-Point Protocol	0.830	15.363	0.004
ATM	0.695	12.488	0.005
Frame Relay	0.850	15.349	0.004

4.3 ATM with E1 link and 64K vs. ATM with E1 link and 8K Nagle and ATM with E1 link and 64K Nagle Algorithm

In this section, the performance of applications (Email, HTTP page download, FTP file download, Remote Login and TCP Delay) are investigated under layer 2 protocol ATM (E1 link and TCP 8K and 64K window sizes) with Nagle algorithm and ATM 64K without Nagle algorithm. The following results are found as per different cases.

4.3.1 Email Response Time

ATM with E1 and 8K TCP Window size with Nagle algorithm have shown better response time than others in case of Email response time (Fig. 11).

The average summary of response times regarding Email download is as following in Table 11.

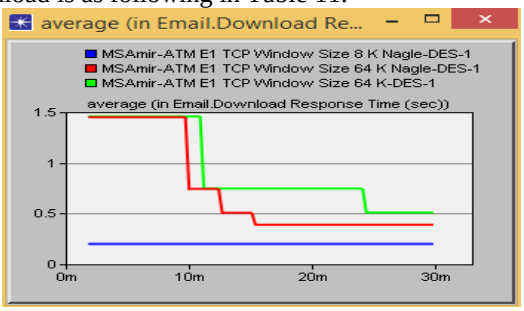

Fig. 11. Email Download Response Time (ATM with E1 link and 64K vs. 8K Nagle, 64KNagle)

Table 11: Email Download Response Time (ATM with E1 64K, ATM with E1 link and 8K Nagle, FRAME RELAY E1 64K Nagle)

ATM Protocol with E1 Link	Email download Response Time (sec)		
	Avg.	Max.	Min.
64K TCP window size	0.5100	1.4575	0.0363
8K TCP window size and Nagle Algorithm	0.2006	0.2006	0.2006
64K TCP window size and Nagle Algorithm	0.3900	1.4507	0.0364

4.3.2 File Transfer Protocol Response Time

In case of File Transfer Protocol, ATM with E1 link and TCP Window size 64K with Nagle and without Nagle has shown better response time than ATM with E1 and TCP Window size 8K Nagle (Fig. 12).
The average summary of response time values regarding FTP download is shown in Table 12.

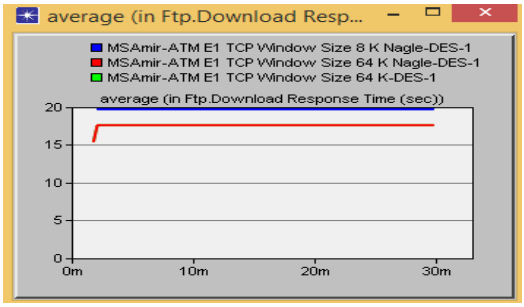

Fig. 12. FTP Download Response Time (ATM with E1 link and 64K vs. 8K Nagle, 64KNagle)

Table 12: FTP Download Response Time (ATM E1 64K, ATM E1 8K Nagle, FRAME RELAY E1 64K Nagle)

ATM Protocol with E1 Link	FTP download Response Time (sec)		
	Avg.	Max.	Min.
64K TCP window size	17.625	19.751	15.498
8K TCP window size with Nagle Algorithm	19.726	19.726	19.726
64K TCP window size with Nagle Algorithm	17.624	19.762	15.486

4.3.3 HTTP Page Download Response Time

ATM with E1 link and TCP Window size 8K with Nagle Algorithm has shown better response times in the case of HTTP web Page response time in Fig. 13.
The average summary regarding HTTP Page download response times is shown in Table 13.

Table 13: HTTP Response Time (ATM with E1 link and 64K, ATM with E1 link and 8K Nagle, FRAME RELAY with E1 link and 64K Nagle)

ATM Protocol with E1 Link	HTTP Page download Response Time (sec)		
	Avg.	Max.	Min.
64K TCP window size	0.1827	1.1545	0.0406
8K TCP window size with Nagle Algorithm	0.0840	0.2318	0.0576
64K TCP window size with Nagle Algorithm	0.3085	1.5598	0.0557

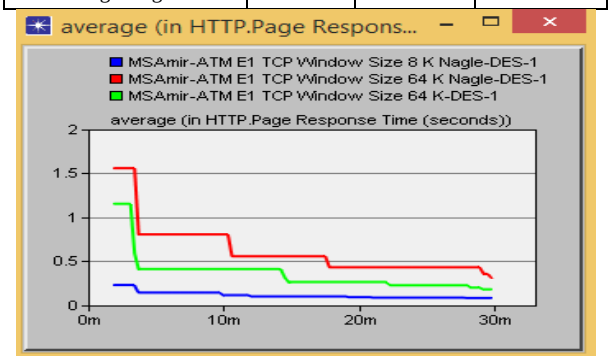

Fig. 13. HTTP Page Response Time (ATM with E1 link and 64K vs. 8K Nagle, 64KNagle)

4.3.4 Remote Login Response Time

ATM with E1 link and TCP Window size 8K Nagle has shown better response time for Remote login when contrast to other technologies (Fig. 14).
The average summary regarding Remote Login response times is shown in Table 14.

Table 14: Remote Login Response Time (ATM with E1 link and 64K, ATM with E1 and 8K Nagle, FRAME RELAY with E1 link and 64K Nagle)

ATM Protocol with E1 Link	Remote Login download Response Time (sec)		
	Avg.	Max.	Min.
64K TCP window size	0.0124	0.0185	0.0094
8K TCP window size with Nagle Algorithm	0.0094	0.0184	0.0094
64K TCP window size with Nagle Algorithm	0.0127	0.0185	0.0094

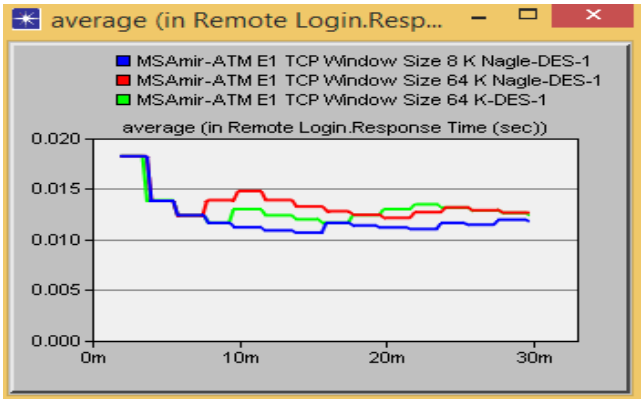

Fig. 14. Remote Login Response Time (ATM with E1 link and 64K vs. 8K Nagle, 64K Nagle)

4.3.5 Transmission Control Protocol Delay

In case of TCP Delay, all links has shown approximately equal delay in OPNET simulation (Fig. 15).

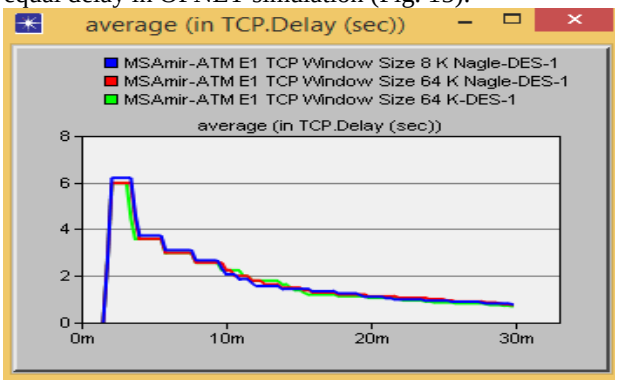

Fig. 15. TCP Delay (ATM with E1 link and 64K vs. 8K Nagle, 64K Nagle)

The average summary regarding TCP Delay response times is shown in Table 15.

Table 15: TCP Delay (ATM with E1 link and 64K, ATM with E1 link and 8K Nagle, FRAME RELAY with E1 link and 64K Nagle)

ATM Protocol with E1 Link	TCP Delay Response Time (sec)		
	Avg.	Max.	Min.
TCP window size	0.695	12.488	0.005
8K TCP window size with Nagle Algorithm	0.785	13.324	0.005
64K TCP window size with Nagle Algorithm	0.755	12.494	0.005

5. Conclusion

In this research, different response times is studied as per scenario designed for different applications including (Email download, FTP file download, HTTP page opening, Remote Login, and TCP Delay) using Point-To-Point Protocol, ATM and Frame Relay (layer 2 protocols) with different TCP window sizes and Nagle algorithm. The conclusion for this research is:

Point-To-Point Protocol (with default TCP window size) shows better response time for the Email download, HTTP page and Remote Login as compared to Point-To-Point Protocol with E1 link and 8K, 32K, 64K TCP window sizes whereas for FTP file download, Point-To-Point Protocol E1 (with 64K TCP window size) shows better results and Point-To-Point Protocol E1 (with 8K TCP window size) shows better results regarding TCP Delay.

When ATM, and Frame Relay are compared with Point-To-Point Protocol (all protocols with 64K TCP window size) for response times of different applications, ATM shows better response times for FTP file download, Remote Login and TCP delay whereas Point-To-Point Protocol shows better results regarding Email and HTTP page download response time. Frame Relay does not show better results in any case.

In the last case, the investigation of response time has been studied using Nagle algorithm with ATM E1 (8K and 64K TCP window sizes) with ATM E1 (64K TCP window size) without Nagle algorithm. ATM (8K TCP window size) with Nagle algorithm shows better response time regarding Email, HTTP, Remote Login applications and ATM (64K TCP window size) with Nagle algorithm is better for FTP file download and ATM (64K TCP window size) without Nagle algorithm is good for TCP Delay.

6. Future Work

For future works, the performance of multimedia applications including voice and video streaming will be conducted as per Transmission Control Protocol network congestion over the wired networks. As this research study is based entirely on wired communication network currently, it will also be considered for wireless communication networks for further works.

References

[1] V. Jacobson and M. J. Karels, “Congestion avoidance and control”, Proceeding of SIGCOMM (Standford, CA) ACM, vol.4, no.18, August 1988.

[2] N. H. Jasem, A. Z. Zukarnain, M. Othman and S. Subramaniam, “Experimental evaluation of ottleneck link utilization with new-additive increase multiplicative decrease” Journal of Computer Science, vol.12, no.5, pp.1058-1062, 2009.

[3] S. Ahmad, A. Mustafa, B. Ahmad, A. Bano and A. Hosam, “Comparative study of congestion control techniques in high speed

network", International Journal of Computer Science and Information Security, vol.6, no.2, pp.222-231, 2009.
[4] F. Akujobi, N. Seddigh, B. Nandy, R. Makkar, and I. Lambadaris, "Congestion control in TCP/IP Networks: A Combined ECN and BECN Approach", MILCOM'03 IEEE conference on Military communications, vol.1, pp.248-254, 2003.
[5] M. A. Talaat, M. A. Koutb, and H. S. Sorour,"Congestion control for internet media traffic", World. Acad. Sci. Eng. Technol., 2008.
[6] Onwutalobi and A. Claret, "TCP congestion control", Codewit News, October 2008.
[7] M. Tekala and R. Szabo, "Throughput analysis of scalable TCP congestion control", Electrical Engineering 51/1-2, pp.57-64, 2007.
[8] G. Minshall, Y. Saito, C. M. Jeffrey and B. Verghese, "Application performance pitfalls and TCP's Nagle algorithm", Center for the study of Language and Information, vol.17, no.15, 1999.
[9] J. Nagle Congestion control in IP/TCP inter-networks, RFC 895, Internet Engineering Task Force, https://tools.ietf.org/html/rfc896, January 1984.
[10] M. Mathis, "Relentless congestion control" Pittsburgh Supercomputing Center, 2009.
[11] H. Sangtae, L. Long, R. Injong and X. Lisong, "Impact of background traffic on performance of high-speed TCP variant protocols", Computer Networks, 51, pp.1748-1762, November, 2007.
[12] C. Parsa and J. Garcia-Luna-Aceves, "Improving TCP congestion control over internet with homogeneous transmission media", Computer Engineering Department, Baskin School of Engineering University of California Santa Cruz, California 95064, 1999.
[13] K. Srinivas, A. A. Chari, and N. Kasiviswanath,"Updated congestion control algorithm for TCP throughput improvement in wired and wireless network", Global Journal of Computer Science and Technology vol.9, no.5, ver.2.0, pp.25-29, 2010.
[14] H. Jamal and K. Sultan, "Performance analysis of TCP congestion control algorithms", International Journal of Computers and Communication, vol.2, no.1, pp.30-38, 2008.
[15] T. Al-Radaei and Z. Zukarnain, "Comparison study of transmission control protocol and user datagram protocol behavior over multi-protocol label switching networks in case of failures", Journal of Computer Science, vol.12, no.5, pp.1042-1047, 2009.
[16] K. Chandrayana, "Comparative study of TCP compatible binomial congestion control schemes", In: B. Sikdar and S. Kalyanaraman ed. Proceedings of IEEE Workshop on High Performance Switching and Routing (HPSR), Kobe, Japan, pp.224-228, May 2002.
[17] K. S. Reddy and C. R. Lokanatha, "A survey on congestion control mechanisms in high speed network", International Journal of Computer Science and Network Security, vol.1, no.8, pp.187-195, 2008.

Muhammad Kashif Hanif is Assistant Professor at the Department of Computer Science at the Government College University, Faisalabad. He received his PhD from Hamburg University of Technology. His research interests cover big data analytic, parallel scientific computing, social media networks, and data mining**.**

Sheikh Muhammad Aamir received the BSc. (Math-A & Math-B) and MSc. (Computer Science) degrees, from Bahauddin Zakariya Univ. in 1999 and 2001, respectively. He received the MSCS degree from Govt. College Univ. in 2011. After working as a Lecturer (from 2002), in Garrison Science College, Barnala AJK, Assistant DBA (from 2004) in MIS Dept. of Pak China Chemicals Lahore, and Network Administrator (from 2008) in IT Services, GC Uni. Faisalabad, he has been a Lecturer in Department of Computer Science at GC Univ. Faisalabad since 2013. His research interest includes Networking, Databases, Machine Learning and computer programming.

Ramzan Talib is Associate Professor at the Department of Computer Science at the Government College University, Faisalabad. He received his PhD from the University of Bayreuth, Germany. His research interests include Databases and Information Systems, Data Mining, Data Warehousing, Business Process Management, Workflow Management Systems.

Muhammad Yahya Saeed received the MSc degree from UAAR, Rawalpindi, in Computer Science in 2003 and M.S degrees in Computer Science from UAF, Faisalabad in 2007. He has also done MSc from UAF, Faisalabad in Statistics in 2000. Currently he is doing PhD from GCUF, Faisalabad in Computer Science since 2015. He is faculty member in GCUF in software Engineering Department.