ACSIJ Advances in Computer Science: an International Journal, Vol. 2, Issue 5, No.6 , November 2013
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

A Survey: variants of TCP in Ad-hoc networks

Komal Zaman[1], Muddesar Iqbal[1], Muhammad Shafiq[1], Azeem Irshad[2], Saqib Rasool[3]

[1]Faculty of Computing & Information Technology
University of Gujrat, Gujrat, Pakistan
komal_zaman@ymail.com,{m.iqbal, shafiq}@uog.edu.pk

[2]Department of Computer Science & Information Technology
International Islamic University, Islamabad, Pakistan
irshadazeem2@gmail.com

saqib.pk@uog.edu.pk
[3]Computer science department,
National University of Science and Technology (NUST), Pakistan

Abstract:
MANET (Mobile Ad-hoc network) forms a temporary network of wireless mobile nodes without any infrastructure where all nodes are allowed to move freely, configure themselves and interconnect with its neighbors to perform peer to peer communication and transmission. TCP (Transmission Control Protocol) offers reliable, oriented connection and mechanism of end to end delivery. This article provides the review and comparison of existing variants of TCP for instance: The TCP Tahoe, The TCP Reno, The TCP New Reno, The Lite, The Sack, The TCP Vegas, Westwood and The TCP Fack. TCP's performance depends on the type of its variants due to missing of congestion control or improper activation procedures such as Slow Start, Fast Retransmission, and Congestion Avoidance, Retransmission, Fast Recovery, Selective Acknowledgement mechanism and Congestion Control. This analysis is essential to be aware about a better TCP implementation for a specific scenario and then nominated a suitable one.

Keywords: Mobile ad hoc network, TCP, Congestion control, Congestion Avoidance

1. Introduction

A major Internet protocol is Transmission Control Protocol (TCP) [1][2] that approximately carries 90% traffic of Internet in today's diverse wireless and wired networks. TCP is end to end reliable protocol that delivers between two objects a consistent data transmission. It is extensively used as an oriented connection of transport layer protocol which offers reliable delivery of data packet over undependable links. The primary goal of TCP is to provide reliable services of data transfer and an oriented connection among different applications to make them able to provide these services on top of an unreliable communication system. Therefore, TCP needs to consider reliability flow control, TCP segments, data transfer, multiplexing, connection management and congestion control. Transmission

Control Protocol does not depend on the underlying network layer which leads to design of several TCP variants based on wired network's properties. On the other hand, congestion control algorithms of TCP may not give sound performance in diverse and heterogeneous environment. Transmission Control Protocol extensively tuned at transport layer to give good performance in old wired network. However, the existing form of Transmission Control Protocol is not well suitable for MANETs where broken routes generates the packet loss cause for TCP's congestion control mechanism's invocation.

Even though many researches and protocol modifications have been directed and recommended. The motive of TCP's variations is to holds some distinct criteria for example:

- Traditional TCP has turn into Tahoe TCP [6].
- Enhances new mechanism by TCP Reno [7] known as Fast Recovery to TCP Tahoe [2].
- TCP New Reno [8] uses TCP Reno's latest retransmission technique [3].
- TCP Sack [9] allows the receiver for the specification of numerous additional out-of-order received data packets [4].
- New retransmission and congestion control schemes proposes by TCP Vegas.
- TCP Fack is basically forward acknowledgement with TCP Reno [5].

This article describes TCP Algorithms and TCP Variants respectively and finally describe comparisons of all variants.

The order of rest of the paper is: Transmission Control Protocol Algorithms and TCP Variants describe by

section 2 & 3 respectively. Section 4 describes the comparisons of all described variants.

2. Algorithms of TCP

Demonstration of different algorithms depending upon slow Start (Exponential Increase), congestion avoidance (Additional Increase), fast retransmission, fast recovery, retransmission, congestion control and selective acknowledgment (SACK) approaches are discussed below.

2.1 Slow Start (Exponential Increase):

Slow Start [10] is a scheme for transmission rate control used by sender. This is also called flow control on sender based. This algorithm is directed for every TCP connection's establishment where maximum available bandwidth is main purpose of it where it can send data without making the network congested. To understand this, TCP sender is forces by slow start to transfer data at a rate of slow sending and quickly increasing it until the existing bandwidth is supposed to be found between the hosts.

A new window to the sender's TCP is provided by this mechanism which is presented in [11] known as the congestion window (cwnd).

Contention Window's size is increases through one segment, every time when an acknowledgment is received that permits the sender for two new segments sending. Thus this approach causes the contention window's exponential growth.

Sometime congestion window become too big for that network which changes the conditions of network such as dropping of packets which causes to generate a sender side timeout and thus the TCP interpret the lost packets as congestion indication and come in congestion avoidance.

2.2 Congestion Avoidance (Additional Increase):

During the stage of data transfer a mechanism of Slow Start is used. During Slow Start, many packets are drop due to congestion. Therefore to slow the rate of transmission a mechanism of Congestion Avoidance is used. A combination of Slow Start and congestion avoidance [10] (two different approaches) is used to do again data transfer which has lost.

TCP indicate jamming by packet loss and through this the mechanism of Congestion Avoidance is invokes by TCP [11].

A new TCP variable (ssthresh) is introduced that means a slow start threshold which is used by TCP to find if there conducted a slow start and congestion avoidance mechanism.

2.3 Fast Retransmission:

In Fast Retransmission [10], if a section is received that is not in an order then Transmission Control Protocol produces duplicate acknowledgment which is instantly sent from receiver to the sender to signifying the arrival of out-of-order segment and also signifying the supposed segment that should be received. In the meantime, this is not necessary to know the causes of the duplicate acknowledgment that may be lost segment or segment's reordering. Therefore before segment's resending, sender have to waits for three duplicate acknowledgments. An advantage of this approach [11] is that for the expiry of retransmission timer the Transmission Control Protocol does not h wait. So for lost segment, three duplicate acknowledgments is an assumption of good sign.

2.4 Fast recovery:

After retransmitted the missing segment, the TCP start the fast recovery [10, 11] algorithm up until a unique acknowledgement comes. The fast recovery is an enhancement of the algorithm of congestion control that even makes the higher throughput sure in adequate congestion and the duplicate acknowledgements are generated by receiver side when another segment is reached to it. Therefore receiver's buffer save this segment and no network resources is consume which means in network the flow of data is running and Transmission Control Protocol is unwilling to move the data in segment of slow start to instantly reduce the flow. Therefore instead the segment of slow start, the congestion avoidance segment is high as soon as the algorithm of fast retransmission is accomplished.

2.5 Retransmission Algorithm:

The Retransmission Algorithm [12] maintains the track of every segment that was transmit and also computes an estimation of the RTT that how much time it takes to get back for the acknowledgment. Every time when there comes a duplicate acknowledgement then this algorithm checks if the transmission time of current time segment is greater than RTT Estimate, then without waiting for 3 duplicate acknowledgements or coarse timeout it instantly resends the segment [12]. Therefore it becomes unable to sense drop packets once it has a small window and can't receive enough duplicate acknowledgements. When a unique acknowledgement is received then resend the missing

segments. If unique acknowledgement (non-duplicate) is the first or second acknowledgement after a fresh acknowledgement then timeout value is check and if the timeout is exceed then without waiting for duplicate acknowledgment it re-transmits the segment [12]. Through this the multiple packet losses sense by TCP Vegas.

2.6 Congestion control Algorithm:

In TCP, four unique congestion control algorithms are used to achieve the Congestion control algorithm [11] in which every algorithm give its best input by influencing from other three algorithms simultaneously.
1. Slow Start (SS) also called operating mode which avoid from presenting congestion.
2. Congestion Avoidance (CA) mode in which without causing under congestion CA make it best to maintain the large amount of data for TCP.
3. Fast Retransmit
4. Fast Recovery
Third mode and fourth mode are very close to each other and almost grouped together which present a solution for long delays in TCP. The last two algorithms help to detect the lost packets and resend them quickly.

2.7 Selective Acknowledgment (SACK):

When TCP miss its acknowledgements then it causes the dropping of multiple segments which also affect the overall throughput by reducing it. Therefore SACK 13] is used to improve this act by updating the sender about every successfully reached segment though receiver. Through this sender only send loss segments. If irregular packet's blocks are received then it also permits the receiver to acknowledge. The number of SACK blocks are also specify by acknowledgment where the starting and ending sequence numbers of an adjoining range is used to convey the SACK block which is correctly receive by the receiver.

3. Evaluation and Description of Variants of TCP

3.1: Evaluation of Variants of TCP

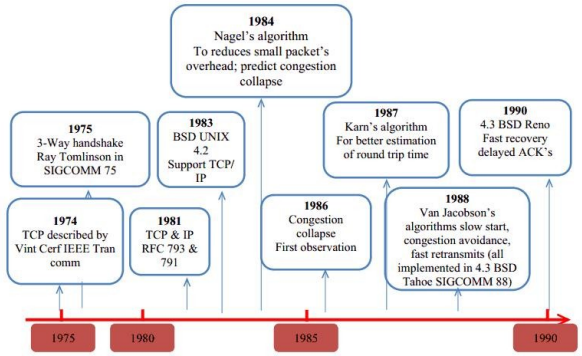

Figure 1: Evaluation of Variants of TCP part (a)

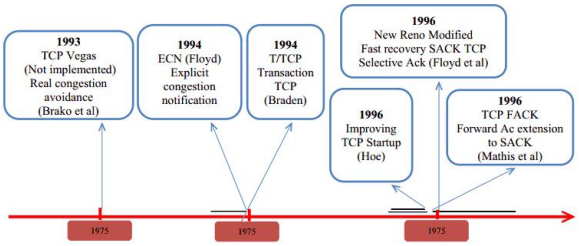

Figure 2:Evaluation of Variants of TCP part (b)

The different variants of TCP are depicted and evaluated in chronological order in Figure1 and Figrue 2.

3.1 Description of variants of TCP

A detailed description on variant of TCP with their problems is providing in this section.

3.2.1 TCP Tahoe:

Tahoe [14] is based on Packet's conservation principle in which the packet should be taken out to insert in the network if a connection is running on the capacity of available bandwidth. It also denotes the algorithm of congestion control. The size of window is 1and TCP will be in phase of slow start when data transmission is start. When timeout the packet loss is detected and window size will become 1. Slow start thresh also become half of current window.
To reflect the network capacity, congestion window also manage by Transmission Control Protocol [3].

- There are some problems that should be solved to gain and maintain the stability.
- Available bandwidth determination.
- Make sure that stability is maintained.
- How to behave when congestion occur?

Tahoe is implemented by adding new and altered procedures such as Slow Start, Congestion Avoidance and Fast Retransmit [6]. An improvement in these algorithms is the modification of round-trip time estimator that sets the value of retransmission timeout [4, 5]. Because of the waiting timeout, Tahoe is not exactly appropriate for product links of high bandwidth.

Problems: For the detection of packet loss Tahoe takes break of complete timeout. Tahoe is unable to send acknowledgment quickly due to cumulative acknowledgements sending which means a "go back n" scheme is follow here. Tahoe waits for timeout and pipeline become empty whenever a packet is drop which introduces high bandwidth delay and a large amount of cost due to it.

3.2.2 TCP Reno

Reno [3, 14] is introduces to overcome the drawbacks of TCP Tahoe. For lost packets detection and don't make empty the whole pipe line when packet drop occur, Reno make some intelligence over Tahoe. It makes the immediate received packet acknowledgement compulsory. Through this it prevents the packet loss. Receiving of Several duplication acknowledgements is the indication of packet loss which means the enough time has been pass even a longer path is taken by data. Therefore an algorithm of Fast retransmission is introduces by Reno which say when there come 3 acknowledgements then consider that packet is lost.
Main algorithm is described below:
Receiving of three duplicate acknowledgements indicate the lost segment. Therefore segment is quickly resent again and enter "Fast Recovery".
"ssthresh" become half of the current window's size and same the value is contention window have.
Increase contention window's value by one on the receiving of every duplicate acknowledgement. If the value of contention window is increases or become high than amount of data in route, then new segment is sent otherwise wait is prefer.

Problems: When packet losses are small then Reno performs fine over TCP. Reno performance becomes unwell when several drop packets fall in one window, because only single drop packet can be sense by it. In case of multiple packet loss, if one report is come about one lost packet then for the acknowledgement of 2nd packet lost it have to wait until reached report of the first segment is come. Another drawback of Reno is of small window size which causes to never receive many duplicate acknowledgements for fast retransmission. Therefore have to wait for a long time. The algorithm of fast recovery is optimized in single packet loss situation.

3.2.3 TCP Lite:

Lite is a facility, offers a transport procedure which disturbs the TCP to diminish overhead exists in session of management where data cannot be received and transmitted. Lite removes the pure TCP protocol's data units used in arrangement and acknowledgement whereas keeping integrity, order, reliability and security of the old TCP. Big window and defense against the wrapped sequence numbers is used by TCP lite.

Problems: The Lite performs above TCP similar to TCP Reno. But when the size of window grows, it has some issues to retain them.

3.2.4 TCP New Reno:

TCP New Reno [15] is a minor alteration above RENO which is able for multiple packet losses detection and
That's why in the case of numerous packet losses it is much efficient than TCP RENO. Similar Reno when New-Reno gets multiple duplicate packets then enters Fast Retransmit. But TCP NEW RENO disables the issue of reducing the CWND faced by Reno several times.
The phase of Fast Transmit is similar as in Reno. There is variance in the stage of Fast Recovery that permits for several Re Transmissions in TCP NEW RENO. Every time it records the maximums segment that is really outstanding when NEW RENO enters in fast recovery. The phase of fast recovery proceeds same like in Reno, conversely when a fresh acknowledgement is got. In this case there are two cases; when we entered fast recovery then if it acknowledges each outstanding segment its means it leavings the Fast Recovery mechanism, fixed CWND is to ssthresh and remains Congestion Avoidance as Tahoe.
If there is incomplete acknowledgement then it assumes that the next segment in the line was vanished, then resends that segment, received duplicate acknowledgement set to zero and exoduses an algorithm

of Fast Recovery when all windows' data is recognized [15].

Problems: New Reno has a downside because one RTT it takes to sense every packet loss. Round-trip Time is used until each lost packets has been retransmitted from the window.

3.2.5 The TCP Sack:

Selective Acknowledgments (SACK) [16] is an extension of Reno which is used to overcome the flaws face by RENO and New-Reno, such as multiple drop packets' detection and per RTT the retransmission of many lost packet. TCP SACK maintains the RENO's slow start and fast-retransmits algorithms. If packet loss is not sensed by the altered mechanism then Sack also has the Tahoe's coarse grained timeout. TCP SACK makes inquiries for the segments that is not cumulatively recognized but should be selectively accepted. Therefore all acknowledgements have a block that defines the recognized segments. Each time when sender move in Fast Recovery then it prepares a flexible pipe that is an estimation outstanding data in the network and set the Contention window to half of the existing size. When the window size goes lesser than Contention window then it checks the non-received segment and transmits that segment. It sends a new packet if no outstanding segments exist [3]. As a result in one RTT many lost segments can be sent.

Problems: TCP SACK needs those segments which are not recognized cumulatively but must be recognized selectively. Every acknowledgement has a block that defines the recognized segments. The major issue of SACK is not the delivery of its presently selective acknowledgements for implementation by the receiver. This also faces the identical problems with multiple losses.

3.2.6 The TCP Vegas:

Vegas [17] are an alteration of TCP Reno, which is dissimilar to TCP Reno in such a way that

- The new Re Transmission approach is used.
- A better Congestion Avoidance algorithm which handles the buffer occupies.
- An improved Slow Start algorithm.
- This solves the problem of coarse gain timeout. Vegas contain an altered retransmission scheme which founded on RTT's fire-gained measurements and new algorithm for Detection of Congestion in Slow Start and Avoidance from Congestion.

Vegas improved the Reno's retransmission algorithm which results in poor estimates due to coarse grained timer use for RTT estimation. Therefore, Vegas save the system clock every time a packet is sent. When there an acknowledgement is get, the Vegas compute RTT and use this for exact approximation decision to resend in the two situations which are described below;

1. When it gets a duplicate Acknowledgement then Vegas checks it to see if RTT is bigger than timeout. If it is greater than tome out, then it quickly resends the packet short of waiting for the third duplicate Acknowledgement.
2. When there a non-duplicate Acknowledgement is received, then after a retransmission if it is the 1st or 2nd Acknowledgement, to check that RTT is greater than timeout Vegas checks it again. If it is greater than TCP Vegas resend the packet.

Problems: If sufficient buffer exist in routers which specify which congestion avoidance algorithm of TCP Vegas can perform greater throughput and result of faster reply time. As burden increases or the number of router buffer decreases, congestion avoidance algorithm of TCP Vegas is not as in effect and start to act like Reno. In use of router buffer TCP Vegas are fewer violent than Reno because TCP Vegas is restricted. In conclusion the congestion detection mechanism of TCP Vegas rest on the correct value for Base RTT.

3.2.7 TCP Westwood:

TCPW is an alteration of sender-side-only to New Reno which proposed towards efficiently grip the product paths of large bandwidth delay with potential packet loss through broadcast. Westwood protocol is based on a simple TCP source protocol's alteration for faster recovery which is implemented by setting the threshold of slow start & values of congestion window which consequence from operative, however congestion is knowledgeable. Therefore, TCP Westwood tries to make a more "knowledgeable" decision in compare with Reno that mechanically splits the congestion window after three duplicate Acknowledgements. Similar to Reno, TCP Westwood cannot differentiate among overflow losses of buffer and the random losses. But, in the existence of random losses, the Reno reacts excessively, thus diminishes the window by half.

Problems: TCPW cannot differentiate between overflow of buffer and the random losses. For data packet or Acknowledgement, TCPW does not provide fast recovery algorithm

3.2.8 TCP Fack:

Improvement in SACK through Forward Acknowledgement is known as TCP FACK [16]. The use of FACK is just about same like TCP SACK but creates a little improvement estimated to it. This uses TCP SACK for efficiently evaluation the data's amount in transit [16]. Thus, TCP FACK announces an improved mode to split the window in case of congestion detection. When Contention Window is split instantly then sender pause sending for some time and then starts again when sufficient data has gone from the network. Here one RTT can be ignored when window is diminished step by step [16].Once there happens congestion then window would be split according to the multiplicative reduction of the accurate Contention Window. In the meantime the sender detects jamming, after it at least one RTT occurred. In between that RTT if it was in slow start manner then present Contention Window almost will be double than the Contention Window when congestion happened. So, in this situation, Contention Window is first split to evaluate accurate Contention Window which further ought to be reduced.

Problems: The Fack delivers congestion avoidance and fast retransmission algorithm, it faces many circumstances for recovery and it can't be easily implement.

4. Comparison of Variants of TCP

4.1 TCP Tahoe:

Tahoe can detect and resend the lost packets quicker than timeouts in Tahoe. This has less re-transmission and does not un-fill the entire pipe when it drops the packets. TCP Tahoe is fine on congestion avoidance and uses the network resources more efficiently due to altered congestion avoidance algorithm and the slow start procedures that calculate arising congestion as well as correctly measures the existing bandwidth. It is not more appropriate for those products links who consume high bandwidth due to waiting timeout.

4.2 TCP Reno:

Vegas banned half timeouts of coarse grained of Reno as it identifies and retransmits many lost packet before the break happens. It can transmit quicker because at all times, it does not have to wait for three duplicate packets. In Reno, the window of congestion does not decrease gradually. It has advantage of congestion avoidance and bandwidth utilization over TCP Tahoe. Due to several packets dropping from the data's window of, it faces the performance issues.

4.3 TCP Lite:

There is not advantage of TCP Lite over TCP Reno; in fact it is same as TCP Reno. It senses and resends many missing packets before the break happens. It suffers the performance problems in face of large amount of dropped packets. TCP Lite offers large window and defense in contradiction of the option of the wrapped sequence numbers that causes the better congestion avoidance and bandwidth utilization and that's why take an advantage over Tahoe and Reno, but similar to Reno TCP Lite does not decrease the congestion window too small for congestion avoidance. In case if there come packet loss in the network then it proposes the better way for fast retransmission.

4.4 TCP New-Reno:

TCP New Reno does not require waiting for 3duplicate acknowledgements before re sending a lost packet and that's why avoids a lot of the coarse grained timeouts. Its congestion avoidance algorithms are very efficient. It also utilizes the network resources in a very efficient way. There are less retransmits due to its altered algorithms of congestion avoidance and slow start algorithm.

4.5 TCP Sack:

There is not such an advantage of TCP Vegas over. TCP Vegas provides better consumption of bandwidth and less significant congestion than SACK. It uses packet loses for congestion indication which make it more stable than SACK.
Therefore sender constantly increases the sending rate up to congestion. There is another disadvantage of SACK is that SACK which is advantage of Vegas that SACK is incorporate in current TCP.

4.6 TCP Vegas:

TCP Vegas offers fast recovery algorithm to overwhelm the packet loss problem and congestion too. It uses better congestion avoidance algorithm which controls the buffer occupies. It uses an improved retransmission schemes that are based on the RTT's Fire Gained measurements to solve the coarse gain timeout.

4.7 TCP West-wood:

TCP Westwood uses Fast Retransmission algorithm to handles dynamic load and huge bandwidth delay routes. When congestion occurs in the network then it uses slow start threshold and algorithm of congestion control.
There is an advantage of TCP Westwood that it has bandwidth utilization and congestion avoidance over network problems.

4.8 TCP Fack:

TCP FACK has an advantage over TCP Westwood by providing a better congestion detection ways in network. This performs estimation for right congestion window which should be reduced further. When the window is gradually reduced then RTT can be avoided. When there detect congestion then TCP FACK presents an improved way to share out the window.

Table no. 1: Comparison of TCP Variants

PARAMETERS	TCP Variants							
	TCP Tahoe	**TCP Reno**	**TCP Lite**	**TCP New-Reno**	**TCP Westwood**	**TCP SACK**	**TCP FACK**	**TCP Vegas**
Congestion Avoidance	Yes	Yes	Yes	Yes	Yes	Yes	Yes	IV
Congestion Control	No	No	No	No	No	No	NA	NA
Slow Start	Yes	Yes	Yes	Yes	Yes	Yes	IV	IV
Fast Recovery	No	Yes	Yes	IV	IV	IV	IV	Yes
Fast Retransmission	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Retransmission	No	No	No	No	No	No	No	NA

5. Conclusion

A concise review of existing TCP variants and their appropriate algorithms are evaluated and define that which protocol is appropriate for the packets, for the utilization of link in the congestion network and the failure of the link causes the disorder in Ad-hoc network because old TCP deals with each packet losses only due to jamming not from the failure of link. This review is achieved and analyzed from the variants of TCP for instance, TCP Tahoe; TCP Reno; TCP New Reno; TCP West-wood; TCP Lite; TCP Sack; TCP Fack and TCP Vegas. Some protocols demonstration their best uses and some shows bad responsiveness to network varying situations and utilization of the network. Even though there are used several protocols and mechanisms but not a single mechanism can be used that can reduces and eliminating the congestion and unreliable network's nature. In solution for the network's problems of TCP protocol, each variant of TCP has its specific advantages and disadvantages. To cut a long story short, simply several protocol will be operative depend on the strictures which are consider as in this review paper. There are many researches available on these variants but still more researches can be done towards the establishment of new protocols. Therefore, this article will help those who want to explore the work done on these variants and want to research more in this area.

References

[1] J. Postel, "Transmission Control Protocol", RFC 793, Sep 1981.
[2] M. Mathis, J. Mahdavi, S. Floyd, A. Roma now.RFC 2018: TCP Selective acknowledgment options, October 1996.
[3] L.S.Brakmo, L.L. Peterson, **"**TCP Vegas: End to End Congestion Avoidance on a Global Internet**"**, IEEE Journal on Selected Areas in Communication, vol. 13, 1995.
[4] K. Fall, S. Floyd **"**Simulation Based Comparison of Tahoe, Reno and SACK TCP**",** 1998**.**

[5] Renaud Bruyeron, Bruno Hemon, Lixia Zhang: "Experimentations with TCP Selective Acknowledgment", ACM SIGCOMM Computer Communication Review, April 1988
[6] Jacobson, V., "Congestion avoidance and control," Proceedings of the ACM Symposium on Communications Architectures and Protocols, Vol. 18, No. 4, pp. 314-329, Stanford, CA, USA, August 16-18, 1988.
[7] Jacobson, V., "Modified TCP Congestion Avoidance Algorithm," end2end-interest mailing list, April 30, 1990. <URL: ftp://ftp.isi.edu/end2end/end2end-interest-1990.mail.>
[8] Hoe, J. C., "Improving the Start-behavior of a Congestion Control Scheme for TCP," Annual conference of the Association for Computing Machinery's Special Interest Group on Data Communication (ACM SIGCOMM '96), pp.270-280, California, USA, August 26-27, 1996.
[9] Mathis, M., Mahdavi, J., Floyd, S. and Romanow, A., "TCP Selective acknowledgement options," IEFT, RFC 2018 (Status Proposed Standard), 1996. <URL: www.rfc-editor.org/rfc/rfc2018.txt>
[10] Suhas Waghmare et. al "Comparative Analysis of different TCP variants in a wireless environment", 978-1-4244-8679-3/11 ©2011 IEEE
[11] W. Stevens, "TCP Slow Start, Congestion Avoidance Fast Retransmit Algorithm", IETF RFC 2001, January 1997.
[12] S.Floyd, T.Henderson "The New- Reno Modification to TCP's fast Recovery Algorithm"RFC 2582, Apr 1999.
[13] Lawrence, S. Brakmo, Student Member IEEE and Larry L. Peterson"TCP Vegas end congestion avoidance on a Global Internet, October 1995.
[14] O. Ait-Hellal, E.Altman "Analysis of TCP Reno and TCP Vegas".
[15] A.Gurtov and S. Floyd, "Modeling wireless links or transport Protocols,"ACM SIGCOMM, April 2004.
[16] B. Qureshi, M. Othman, Member, IEEE, and N. A. W. Hami "Progress in Various TCP Variants", IEEE, February 2009.
[17] M. Mathis, J. Mahdavi,"Forward Acknowledgement: Refining TCP Congestion Control" in Proceedings of ACM SIGCOMM, 1996.

Ms.Komal Zaman is studying as research student under the enrollment of MS (IT) program in University of Gujrat, Pakistan since 2013. She did graduation with first division in the field of Information Technology from university of Punjab, Pakistan in 2008. She has been working as Associate lecture in university of Gujrat. Her research interest spans the area of QoS issues and routing challenges, especially in mobile ad-hoc networks.

Dr. Muddesar Iqbal has done PhD from Kingston University UK in the area of Wireless Mesh Networks in 2009. He has been serving as Associate Professor in Faculty of computing and Information technology, University of Gujrat, Pakistan since 2010. He won an Award of Appreciation from the Association of Business Executive (ABE) UK for tutoring the prize winner in Computer Fundamentals module. He also received Foreign Expert Certificate from State Administration of Foreign Experts Affairs, People's Republic of China in 2008 against his research collaborations in China. He won another Award of Appreciation from ABE UK for tutoring the prize winner in Information System Project management module in 2010. He has published more than 20 papers, all in International Journals and proceedings. His research interests span the area of mobile ad hoc routing and security issues, analysis and control of wireless networks, resource management including packet scheduling, and wireless sensor networks.

Mr.Muhammad Shafiq did MS (CS) from UIIT, PMAS Arid Agriculture University Rawalpindi, Pakistan in 2010. He received MIT. degree from University of the Punjab in 2006. He has been serving as Lecturer in Faculty of Computing and Information Technology, University of Gujrat, Pakistan since, 2010. He also served as visiting lecturer in the Federal Urdu University, Islamabad, Pakistan for the period of one year. He has published more than 10 papers, in national and International Journals and proceedings. His research interests span the areas of routing, security, QoS, resources and mobility management issues of communication networks, especially MANETS and VANETS.

Azeem Irshad has been doing Ph.D from International Islamic University, Islamabad, after completing MS-CS from PMAS Arid Agriculture University Rawalpindi and is currently serving as visiting lecturer in AIOU. He has published more than 10 papers, in national and International Journals and proceedings. His research interests include NGN, IMS security, MANET security, merger of Ad hoc networks with other 4G technology platforms, multimedia over IP and resolving the emerging wireless issues.

Mr.Saqib Rasool recently has done MS (IT) from National University of Science and Technology (NUST) in 2013 and did his graduation in computer Science from University of Gujrat in 2010. He has been serving as Associate Lecturer in Faculty of Computing and Information technology, University of Gujrat, Pakistan since 2011. He also served as professional developer in CONNEKT labs of SEECS-NUST. He also served in the research project of Semantic Web Application Firewall (SWAF) of NUST. His research interest spans over the area of research and development of communication and distributed systems.