International Journal Of Engineering Research & Management Technology ISSN: 2348-4039

May- 2014 Volume 1, Issue 3

Email: editor@ijermt.org Website: ijermt.org

A Survey on Various TCP Variants In Mobile Ad-Hoc Networks

Amit Kumar,Arwinder Singh
Department of Computer Science & Engineering
Shree Siddhivinayak Group Of Institutions,
Bilaspur, Yamuna Nagar, Haryana, India

ABSTRACT –
In this paper we describe the various TCP variants. There are various variants - TCP, TCP Reno, TCP Newreno, TCP Tahoe, TCP Sack, TCP Fack, TCP Vegas ,TCP Lite and TCP West-Wood, these implemented in network simulator NS-2. Among all these variants the three variants are considered as important for analysis, namely- TCP Reno, TCP NewReno and selective Acknowledgment (SACK). In wired and wireless network TCP used transport protocol. TCP is connection-oriented, reliable and end to end transport protocol. The behavior of TCP depends on the TCP variants.

KEYWORDS- MANETs, TCP variants.

INTRODUCTION

The collection of mobile nodes that dynamically form a network without any network infrastructure is called MANET. In this network nodes communicate with its neighbors to perform peer to peer communication and transmission. There is an immediate communication among neighboring devices in MANETs however communication between non-neighboring devices needs a routing rule. Lots of work has been done on routing protocols since they are crucial to the functioning of ad-hoc networks inside the two classes of routing protocols delineate in literature: Proactive and Reactive, it's a lot of suited to extremely mobile adhoc networks as a result of its ability to address rapidly ever-changing network topologies, as a result of there's no coordination or configuration before setup of a manet, there are many challenges and also these challenges embody routing packets in associate setting wherever the topology is ever-changing frequently and the task of locating a node and maintain a path to that becomes more and more within the face of node quality.

TCP is transport layer protocol, which offers connection-oriented, reliable, end to end transport protocol and byte-stream services. TCP provide stable and reliable transfer of packet data across the internet [6]. Now-a days, most of the internet traffic is carried out as well as the majority of widely used applications are provided by TCP [10], like File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP) make use of TCP of TCP/IP suite for their operation [10]. TCP also provides division for sequenced data stream into packets, confirms the packets delivery with the possibility of losing the IP layer loses, retransmit, reorders, or packets duplication and monitoring the network band capacity to avoiding congestions [1]. There are two variables – congestion window size (cwnd) and SSthreshold (SSthresh) and two distinct phases that are the slow start and congestion avoidance phase.

Slow start- Today in TCP implementation every TCP connection starts off in the "slow start" phase. The new variable which is used by slow start is called congestion window (cwnd). Slow start adds one packet per ACK.

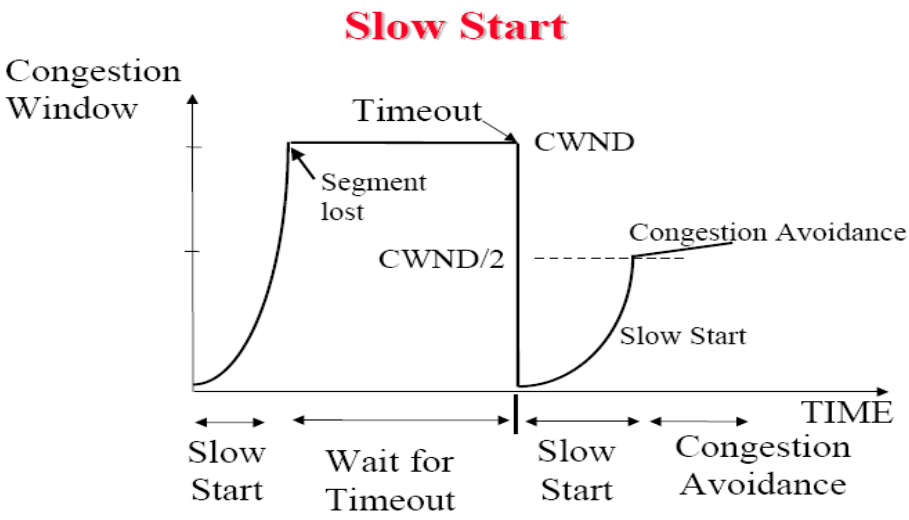

Figure 1: Slow Start

Congestion avoidance- When the number of packets dropped more in the network due to the overload and congestion, then congestion avoidance is used. It is used to slow the transmission rate. In the congestion avoidance phase, the cwnd is increased by 1 full-sized segment every round-trip time (RTT) [7].

In TCP there are three control mechanisms- flow control, error control and congestion control [4]. The flow control defines the amount of data that send before receiving the ACK from destination [4]. For error control three simple tools are used- checksum, acknowledgement and time-out. There are two ways to detect congestion –

- As the result of time timeout
- Receipt of duplicate acknowledgement

If the detection is done using the retransmission timer timeout, the value of ssthresh is updated as follows [7]:
ssthresh = max (Flight Size / 2, 2*MSS)

Figure 2 define the overall operation of TCP

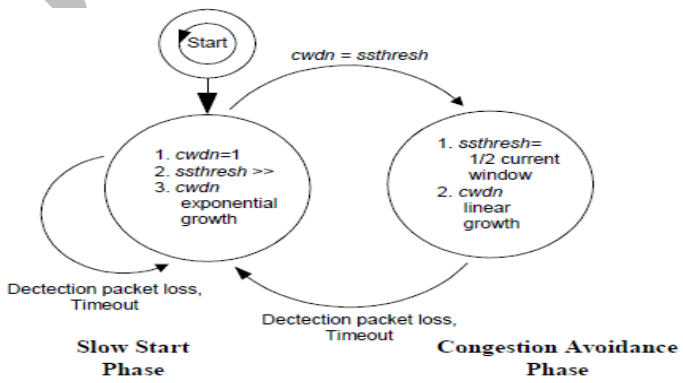

Figure 2: Transition Diagram of TCP

TCP Variants-

1. **TCP Reno**

TCP Reno was developed in 1990.Reno retains the fundamental principle of Tahoe, like slow starts and also the coarse grain re-transmit timer. However it adds some intelligence over it, in order that lost packets detected earlier and also the pipeline isn't empty whenever a packet is lost. In Reno it is necessary to receive immediate acknowledgement when a segment is received. TCP Reno suggests an algorithm called "Fast Retansmit". Reno requires that we receive immediate acknowledgement whenever a segment is received [8]. Reno is applied numerous algorithmic rule to regulate the network congestion that consists of 4 phases; slow start, congestion avoidance, fast retransmit and fast recovery. NewReno is tried to exploiting the losses in packets to deciding the prevailing information measure capability within the network.

It begins slow start procedure within the transmission control protocol affiliation starting also as once timeouts inside affiliation. During this progression it primarily growths exponentially the congestion window and linearly once reaches ssthresh level to begin the opposite section celebrated by congestion avoidance. When timeout happens or if three duplicate ACKs area unit received, fast transmit and fast recovery is timeout interruption to point the congestion in network. The congestion management of Reno doesn't decrease the transmission rate except if it notes a dropping in packet which can happen given that network suffer from overload scenario. Where Reno is attempt to reconciliation the dimensions of window for various connections.

Problem- multiple packet loss in one window

2. **TCP NewReno**

TCP NewReno was developed in 1996. It is a version of TCP Reno supported by some adaption and includes fast recovery mechanism [1], means it is slight modification over TCP Reno. In this multiple packet losses are detect more efficiently than the Reno. Like Reno, TCP NewReno also enter into the Fast-retransmit when it receive multiple duplicate packets, however it differs from RENO in that it doesn't exit fast recovery until all the data which was out standing at the time it

Entered fast recovery is acknowledged [8]. The fast recovery part yield as in Reno, but once a recent ACK is received then there are two cases-

- If it ACK's all the segments that were outstanding once we entered fast recovery then it exits fast recovery and sets CWND to threshold value and continues congestion avoidance like Tahoe.
- If the ACK may be a partial ACK then it deduces that subsequent phase in line was lost and it re-transmits that phase and sets the quantity of duplicate ACKS received to zero. It exits fast recovery once all the information within the window is acknowledged.

3. **TCP Tahoe**

TCP Tahoe introduced by Jacobson in 1998. The basic principle on which it is based- "conservation of packets" i.e if the connection is running at the available bandwidth then a packet is not rejected into the network unless a packet is taken out as well [2]. TCP Tahoe also maintain congestion window (cwnd) to reflect network capacity. In Tahoe congestion control, connections permanently are drive to slow start phase for each losses in packets and when the size of window is big and the loss are infrequent, it's well for connections to start from congestion avoidance phase, due to it will need a time to growing the size of the window from 1 to reaching the value of ssthresh [1]. TCP Tahoe maintains a congestion window CWND to reflect the network capacity [2].

Problems- 1.to detect a packet loss TCP Tahoe take a complete timeout interval, in some cases it takes more time because of the course grain time out [2].
2.it doesn't send immediate ACK's it send cumulative acknowledgement ,means it follow a go-back n approach [2].

4. TCP Vegas

TCP Tahoe was introduced by Brakmo et at. It uses a sophisticated bandwidth estimation scheme [7]. TCP Vegas is the modification of Reno. TCP Vegas proposes its own unique retransmission and congestion control strategies [5]. The differences of Reno and Vegas are-

- A new retransmission mechanism is employed
- Associate degree improved congestion turning away mechanism that controls buffer occupy
- A changed slow begin mechanism

TCP Vegas dynamically varies its congestion window size supported fine-grained measuration of RTTs, whereas TCP Tahoe continues to extend its window size till packet loss is detected [7].
That solve the matter of coarse gain timeout transmission control protocol Vegas embrace a changed retransmission strategy that's supported fire-gained measurements of the RTT (means outlined by system clock) similarly as new mechanism for congestion detection throughout slow start and congestion avoidance. In transmission control protocol city, a rather coarse grained timer is employed to estimate the RTT and also the variance, which ends up in poor estimates. Vegas extend Reno's retransmission mechanism as follows. As mentioned before, Vegas record the system clock on every occasion a packet is distributed. Once an ACK is received, Vegas calculate the RTT and use this additional correct estimate to determine to channel within the following 2 situations:

1. Once it receives a reproduction ACK, Vegas checks to see if the RTT is larger than timeout. If it is, then while not anticipating the third duplicate ACK, it instantly retransmits the packet
2. Once a non-duplicate ACK is received, if it's the primary or second ACK once a retransmission, Vegas once more checks to visualize if the RTT is larger than timeout. If it is, then Vegas conduct the packet.

Problems:
If there are unit enough buffer within the routers it implies that Vegas congestion avoidance mechanism will operate effectively the next output and a quicker reaction time result. Because the load increase or the quantity or router buffer decrease,Vegas congestion avoidance mechanism isn't as effective and Vegas begin to behave a lot of like reno. Vegas is a smaller amount aggressive in its use of router buffer than urban center as a result of Vegas is restricted. Finally Vegas congestion detection rule depends on the correct value for Base RTT.

5. TCP with Selective Acknowledgments (SACK)

TCP with 'Selective Acknowledgments' is an extension of TCP Reno [4] and it works round the issues face by TCP RENO and TCP New-Reno, particularly detection of multiple lost packets and re-transmission of more than one lost packet per RTT [4]. SACK retains the slow-start and fast retransmits parts of RENO [2]. The use of Sacks permits the receiver to specify several additional data packets that have been received out-of-order within one dupack, instead of only the last in order packet received [5]. There is a variable that maintains by SACK called pipe which indicates the number of outstanding transit segments. When the sender sends a new

segment or retransmitted then value of pipe is incremented by one and the value is decremented by one when the sender receives a duplicate ACK with SACK which showing new data has been received.
Dissimilar to Tahoe, with difficulties of the phases of slow begin and congestion avoidance and reno, with irregular performance that happens if multiple dropping in packets in same window of data, protocol Sack performs a lot of direct, simply to understanding and additionally easier to expect. If Sack doesn't use with Reno, it suffers from issues if multiple dropping in packets occur in same window of data and these issues outcome from the need to expect the timer of expiration for retransmission before deciding to resend data. Sack represents an growth of TCP's Reno and NewReno and its operating close to the risks that is moon-faced these 2 variants once multiple packets losses happen and retransmissionof multiple lost packets for every RTT. Once Reno and NewReno congestion control rule doesn't support SACK, they're ready to resend only 1 packet that dropped for every RTT, even once protocol sender recuperate for multiple drops in data window and no have to be compelled to wait the timeout. Additionally, these characteristics doesn't enclosed in Tahoe, whenever isn't any border to resending at greatest single dropped packet for every RTT.

Problem- The biggest downside with SACK is that presently selective acknowledgements aren't provided by the receiver. To implement SACK we'll need to implement selective acknowledgement that isn't a really easy task.

6. TCP Lite

TCP lite could be a service that has a transport methodology that interrupts transmission control protocol so as to scale back the overhead involved in session management during which no data is transmitted or received. Transmission control protocol lite reduces or eliminates pure transmission control protocol protocol data units (PDUs) employed in the established and ACK whereas maintaining order, integrity, reliableness and security of first transmission control protocol. Transmission control protocol lite uses huge window and protection against wrapped sequence numbers. A TCP-Lite transport is applied to a transport profile, that is applied to an affiliation profile. The affiliation profile could be a set of configuration properties appointed to associate MNC to regulate the performance choices between associate MNC and quality purchasers that hook up with it. It is similar to TCP Reno. It detects and re-transmits more than one lost packet before timeout occurs [3].

Problems: TCP Lite perform over transmission control protocol same as Reno. However once window will increase it have some issues to maintain them.

7. TCP Westwood

The protocol Westwood (TCPW) could be a sender-side-only modification to protocol NewReno that's meant to higher handle giant bandwidth-delay product methods, with potential packet loss due to transmission or alternative errors, and with dynamic load.TCP Westwood protocol depends on a straightforward modification of the protocol supply protocol behavior for a faster recovery. This can be performed by setting each a slow start threshold and a congestion window values that result from the effective affiliation whereas congestion is experienced. Hence TCPW makes an attempt to create a more "informed" call, in distinction with protocol reno, that mechanically halves the congestion window once three duplicate ACKs. Like protocol Reno, TCPW cannot distinguish between buffer overflow losses and random losses. However, in presence of random losses, protocol Reno over reacts and reduces the window by 0.5.

Problems: TCP West Wood cannot distinguish between buffer overflow and random losses [2]. It doesn't offer fast recovery mechanism for information packet or ACK.

8. TCP FACK

TCP with forward acknowledgement (FACK) is a different algorithm, that works on upper option of TCP SACK. The FACK option provides an improvement in performance in the case of multiple losses in a single window of data and reduces the overall burstiness of TCP [9]. In FACK, TCP maintains 2 variables that are-(i) fack, that represents the forwardmost segment that has been acknowledged by the receiver through the SACK option (ii) retran_dam, that reflects the amount of outstanding retransmitted data in the network. By using these two variables, the amount of outstanding data during recovery can be estimated as forward-most data sent - forward-most data ACKed (fack value) + outstanding retransmitted data (retran_data value) [11].
To trigger fast retransmit the fack variable is used.
TCP FACK introduces a far better thanks to halve the window once congestion is detected. Once CWND is instantly halved, the sender stops transmission for a short time then resumes once enough knowledge has left the network. During this one RTT will be avoided once the window is step by step attenuated .When congestion occurs; the window ought to be halved according to the increasing decrease of the proper CWND. Since the sender identifies congestion a minimum of one RTT when it happened, if throughout that RTT it absolutely was in slow start mode, then this CWND are going to be nearly double than CWND once congestion occurred. Therefore, during this case, CWND is first halved to estimate the proper CWND that ought to be any attenuated.

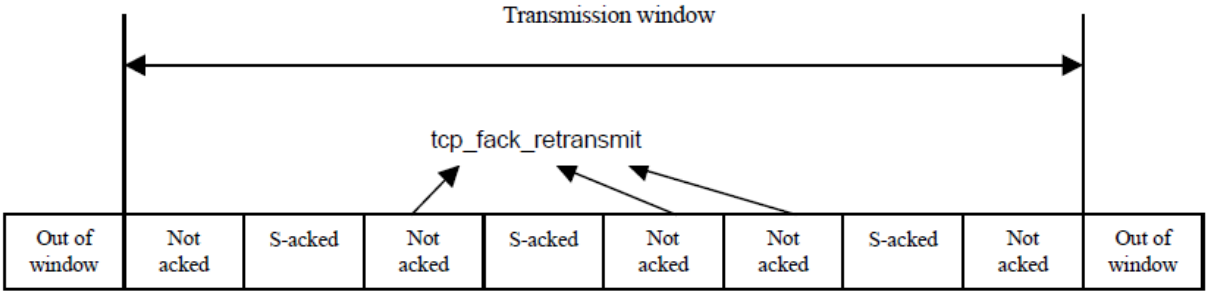

Figure 3: FACK

CONCLUSION

A detail review of existing transport protocol variants and its applicable algorithm measure analyzed and describe regarding the protocol that one is best and appropriate for packet and link utilization within the network congestion and link failure condition in Ad-hoc network setting as a result of the standard communications protocol treat all packet losses as a result of the congestion, it doesn't treat from the link failure. The review square measure obtained and analyzed by the transport protocol variants: transport protocol Tahoe, TCP Reno, communications protocol NewReno, communications protocol West-wood, TCP Sack, TCP Fack, and communications protocol Vegas. The foremost of protocol shows higher uses and lots of them shows poor responsiveness to dynamical network conditions and network utilization. Though there square measure varied protocols and algorithms are used, there's no single rule that may overcome the full and unreliable nature of network. Here each and every variant has its own advantages and drawbacks to unravel the networks downsides of TCP protocol.

TABLE1: Comparison of TCP Variants [2]

Algorithms/Variants	TCP Reno	TCP New Reno	TCP Tahoe	TCP SACK	TCP FACK	TCP Vegas	TCP Westwood	TCP Lite
Slow Start	Yes	Yes	Yes	Yes	E V	Yes	Yes	E V
Congestion Avoidance	Yes	Yes	Yes	Yes	Yes	E V	Yes	Yes
Fast Retransmit	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Fast Recovery	Yes	E V	No	E V	E V	Yes	E V	Yes
Selective ACK	No	No	No	Yes	Yes	No	Yes	No
Congestion Control	N	N	N	N	N M	N M	N	N
Retransmission	N	N	N	N	N	N M	N	N

REFERENCES

1. Ghassan A. Abed, Mahamod Ismail and Kasmiran Jumari, "A Survey on Performance of Congestion Control Mechanisms for Standard TCP Versions", in Australian Journal of Basic and Applied Sciences, 5(12): 1345-1352, ISSN 1991-8178, 2011
2. Mandakini Yayade and Sanjeev Sharma, "REVIEW OF DIFFERENT TCP VARIANTS IN AD-HOC NETWORKS, in International Journal of Engineering Science and Technology (IJEST), ISSN: 0975-5462 Vol. 3 No. 3 March 2011
3. Poonam Tomar and Prashant Panse, "A Comprehensive Analysis and Comparison of TCP Tahoe, TCP Reno and TCP Lite", in International Journal of Computer Science and Information Technologies , Vol. 2 (5) , 2011
4 Heena Dave, Vikas Gupta and Parul Dihulia, " Performance Comparison between TCP Sack and TCP Vegas using NS-2 Simulator", in International Journal of Computer Applications (0975 – 8887) Volume 68– No.11, April 2013
5 Md. Shohidul Islam, M.A Kashem, W.H Sadid, M. A Rahman, M.N Islam and S. Anam, "TCP Variants and Network Parameters: A Comprehensive Performance Analysis", in International MultiConference of Engineers and Computer Scientists, Vol-I IMECS 2009, March 18 - 20, 2009
6 Mazleena Salleh and Ahmad Zaki Abu Bakar, "Comparison of TCP Variants over Self Similar Traffic" in Proceedings of the Postgraduate Annual Research Seminar 2005.
7 S. R. Biradar, Subir Kumar Sarkar and Puttamadappa C, "A Comparison of the TCP Variants Performance over different Routing Protocols on Mobile Ad Hoc Networks", in International Journal on Computer Science and Engineering (IJCSE), Vol. 02, No. 02, 2010
8 Hrituparna Paul, Anish Kumar Saha, Partha Pratim Deb and Partha Sarathi, "Comparative Analysis of Different TCP Variants in Mobile Ad-Hoc Network" in International Journal of Computer Applications (0975 – 8887) Volume 52– No.13, August 2012
9 Giacomo Morabito, Renato Narcisi, Sergio Palazzo and Antonio Pantò, "TCP-Peach and FACK/SACK Options: Putting the Pieces Together".
10 Md Nazmul Islam Khan, Rashed Ahmed and Md. Tariq Aziz, "A SURVEY OF TCP RENO, NEW RENO AND SACK OVER MOBILE AD-HOC NETWORK", in International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.1, January 2012
11 Sonia Fahmy and Tapan Prem Karwa, "TCP Congestion Control: Overview and Survey Of Ongoing Research" in Purdue University Purdue e-Pubs, 2001

IJERMT