International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 02 | Feb 2019 www.irjet.net p-ISSN: 2395-0072

Analysis of Various TCP Variants over MANET Routing Protocols

Deeksha Kotian[1], Shibani Shetty[2], Shifana Begum[3], Akhilraj V Gadagkar[4]

[1,2]U.G Student, Dept of CS&E, Srinivas School of Engineering, Mukka
[3,4]Assistant Professor, Dept of CS&E, Srinivas School of Engineering, Mukka

---

Abstract - *The TCP protocol was developed to supply reliable end-to-end delivery of packets below variable degrees of congestion within the network. They are widely developed for wired networks. As errors usually occur over wireless links the implementation has been tested with variety of emulated wireless networks. The traditional TCP is that the solely higher resolution to handle the problems in wireless networks and compatible with wired network as well. Congestion control in all TCP variants does not show similar performance in MANET as in wired network because of the fault detection of congestion. In this paper, we do a performance comparison between TCP variants such as TCP Tahoe, The TCP Reno, The TCP New Reno, The Fack, The Sack, The TCP Vegas, Westwood and The TCP Lite in DSDV, TORA, AODV and DSR reactive routing protocols this is carried out using NS-2 simulation and it analyze the results with respect to throughput, jitter, packet loss, signal received with error and byte received to see if there is an overall economical TCP variant.*

Key Words: **MANET, NS-2 Simulator, Routing Protocols, TCP Variants.**

1. INTRODUCTION

The Mobile Ad hoc Network (MANET) is widely used nowadays, and it's thought of as a hot topic in network field. In MANET, the nodes can flow freely and at any time it can connect to different nodes. Every node in MANET works as router and client. Mobile Ad-hoc Networks are self - configuring networks consisting of mobile nodes that are communicating through wireless links like radio or microwave with no fastened infrastructure to manage communication between them. Nodes in MANET are moving randomly rather than wishing on the router in coordinative the flow of packets within the network, they send packets to every alternative. Ad-hoc routing protocols are used to determine routes packets should travel through. Each node senses the broadcasting from its neighbors to determine the connection. [1].

Nodes are mobile thus topology keeps on ever-changing. They are applicable in such things wherever no infrastructure is on the market. TCP has established to perform dependably in ancient wired and stationary networks wherever the most reason for losses in network congestion however it doesn't perform as thus once applied to wireless networks. It is due to the misunderstanding of the losses that are not caused by network congestion. Sadly it invokes a congestion control algorithmic program that reduces the bandwidth utilization and become the rationale for performance degradation by providing poor throughput and better delays. When there are many resources in an exceedingly network that are shared by multiple competitive senders it becomes tough to manage the information rate employed by every sender so network needn't be full. The network congestion will cause severe degradation of output. If no correct approach is followed for dominant the congestion than it will even collapse the network. [10].

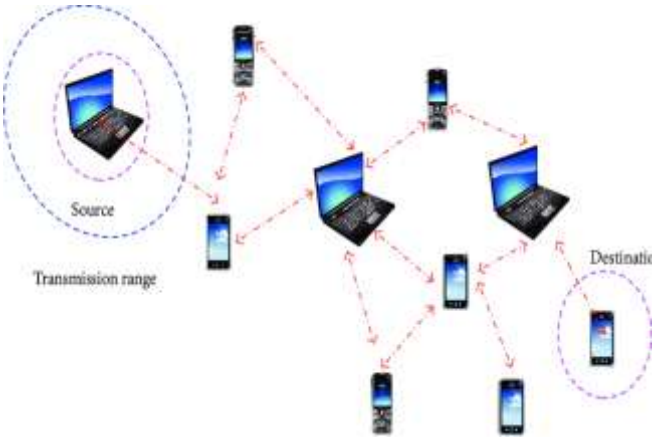

Fig -1: Mobile Ad hoc Network

1.1 Description of TCP Variants

1. Tcp Tahoe

Tahoe is that the simplest TCP variant. It doesn't have quick recovery state and through congestion avoidance phase, Tahoe treats a triple duplicate ACKs same as a timeout[1]. It had been the changed version of basic TCP protocol. Tahoe is enforced by adding new and altered procedures like Slow begin, Congestion avoidance and Fast Retransmit [4]. Associate improvement in these algorithms is that the modification of round-trip time calculator that sets the worth of retransmission timeout [2,3]. Due to the waiting timeout, Tahoe isn't precisely applicable for product links of high information measure. There are certain problems, which require to be resolved to make sure this equilibrium.

1) Determination of the offered bandwidth.

2) Making certain that equilibrium is maintained.

3) A way to react to congestion.

Problems: the matter with Tahoe is that it take an entire timeout interval to observe a packet loss and of course, in most implementations it takes even longer thanks to the coarse grain timeout. Additionally since it does not send immediate ACK's, it sends accumulative acknowledgements, so it follows a 'go back n' approach. Tahoe waits for timeout and pipeline become empty whenever a packet is drop that introduces high bandwidth delay and an oversized quantity of price due to it.

2. Tcp Reno

Reno has same congestion control algorithms as Tahoe with associate addition of quick recovery. In quick recovery, ssthresh and new cwnd is ready to half this cwnd rather than setting the cwnd to one. So Reno skips slow start and directly enters congestion avoidance [5].

Fast Recovery is listed by a protocol sender later obtaining associate initiative threshold of duplicate acknowledgement. This threshold code is typically fastened to three. Once the threshold of duplicate acknowledgement is taken, the sender retransmits one packet and minimize its congestion window by one divide. Rather of slow start section, as is accomplish by a protocol Tahoe sender, the Reno sender want to further duplicate acknowledgement to clock later outgoing packets. Reno Fast Recovery algorithmic rule is optimized because the state of affairs once a particular packet is discarded from a window of data. The Reno sender retransmits just one discarded packet per round trip time.

TCP Reno extraordinarily enhance upon the action of protocol Tahoe whenever a particular packet is discarded from a window of data packet however will endure with performance issue when many packets are dropped from a window of data packets. This issue is definitely designed in our simulator throughout a protocol Reno connection with a big congestion window deteriorate a burst of packet deficit once slow-beginning in an exceedingly connection with drop-tail gateways or further gateways that decline to guide traditional the queue capability [7].

Problems: Reno performs very well over TCP once the packet losses are small. However once we have multiple packet losses in one window then RENO doesn't perform too well and its performance is nearly the identical as Tahoe below conditions of high packet loss. The reason is that it will solely discover one packet losses. If there is multiple packet drop then the primary data concerning packet loss comes once we receive the duplicate ACK's. But the information about the second packet that was lost can come back solely once the ACK for the retransmitted initial phase reaches the sender after one RTT.

3. Tcp New Reno

TCP New Reno is a conversion of TCP Reno. It is capable to search out once many packet losses are arising within the network. It is more practical than Reno in the case of several packet losses is arises in the computer network. New Reno enhances retransmission as the fast reformation phase of TCP Reno. New Reno TCP is an improved version of Reno that avoids multiple degradation of the congestion window whereas multiple segments from the identical window of data get lost[7].

The congestion control techniques for TCP New Reno are just like TCP Reno except the fast recovery phase of TCP Reno had improved. Like RENO, New-RENO also enters into fast-retransmit once it receives multiple duplicate packets, but it differs from RENO in this it doesn't exit fast-recovery till all the information that was out standing at the time it entered fast recovery is acknowledged. The fast recovery [6] phase proceeds as in Reno, however when a fresh 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 CWD to threshold value and continues congestion avoidance like Tahoe.
- If the ACK maybe a partial ACK then it deduces that the following phase in line was lost and it re-transmits that segment and sets the quantity of duplicate ACKS received to zero. It exits Fast recovery when once all the information within the window is acknowledged.

Problems: New-Reno suffers from the actual fact that its take one RTT to detect every packet loss. Round-trip Time is employed till every lost packets has been retransmitted from the window. Once the ACK for the primary retransmitted segment is received solely then will we tend to deduce that alternative segment was lost.

4. Tcp SACK

SACK is associate extension of Reno. Another way to deal with multiple segment losses is to inform the sender that segments have arrived at the receiver. Selective Acknowledgments (SACK) TCP[8] does exactly this. The receiver uses every TCP SACK block to point to the sender one contiguous block of data that has been received out of order at the receiver. Once a SACK blocks are received by the sender, they are accustomed maintain a picture of the receiver queue, i.e., that segments are missing and which have created it to the receiver. Using this information, the sender retransmits solely those segments that are missing, while not looking ahead to a retransmission timeout. Only no section has to be retransmitted, new data segments are sent out.

In SACK, selective ACKs are done rather than cumulative ACKs. Each ACK has a section which contains the sequence number of packets that have been acknowledged. Once TCP enters fast recovery, SACK implements a parameter named "pipe" that represents the estimate of the amount of unacknowledged packets in network. The congestion window is reduced to half of the current window. For every

ACK received pipe is decremented by 1 and for every packet retransmitted pipe is incremented by 1. If there are no un-ACKed packets within the network a new packet is transmitted. Thus, using SACK multiple packets can be retransmitted in precisely one RTT[5,6].

Problems: TCP SACK desires those segments that are not recognized cumulatively however should be recognized by selection. Every acknowledgement includes a block that defines the recognized segments. To implement SACK we will need to implement selective acknowledgment that isn't a really simple task.

5. Tcp FACK

It is an acronym for TCP Forward Acknowledgement(FACK). It was the modified version of Sack. It will facilitate TCP to survive multiple phase loses inside one window without acquire a retransmission timeout. FACK[9] also aims at better recovery from multiple losses. The name "forward ACKs" comes from the actual fact that the algorithmic program keeps track of the properly received data with the best sequence range. In FACK, TCP maintains a pair of extra variables: (1) fack, that represents the forward most segment that has been acknowledged by the receiver through the SACK choice, and (2) retran_data, that reflects the quantity of outstanding retransmitted data in the network. Using these 2 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).

Problems: The Fack delivers congestion rejection and fast retransmission algorithm, it faces several circumstances for recovery and it can't be simply implemented.

6. Tcp Vegas

TCP Vegas was conferred in 1994 before New Reno, SACK and FACK were developed. It had been enforced by Larry Peterson. During this timeouts were set and round-trip delays were measured for each packet within the transmit buffer.TCP Vegas uses additive will increase construct within the congestion window.

The three major changes induced by Vegas are:

- New Re-Transmission Mechanism: Vegas extend on the retransmission mechanism of RENO. It keeps track of once every section was sent associated it additionally calculates an estimate of the RTT by keeping track of however long it takes for the acknowledgment to induce back.
- Congestion avoidance: TCP Vegas is completely different from all the other implementation in its behavior throughout congestion avoidance. It doesn't use the loss of section to signal that there's congestion. It does not use the loss of segment to signal that there is congestion. It determines congestion by a decrease in sending rate as compared to the expected rate, as result of large queues building up in the routers.
- Modified Slow-start: TCP Vegas differs from the other algorithms throughout its slow-start section. The explanation for this modification is that once a connection first starts it has no plan of the accessible bandwidth and it is possible that in exponential increase it over shoots the bandwidth by an enormous quantity and so induces congestion. To the present finish Vegas will increase exponentially solely each different RTT, between that it calculates the particular sending throughput to the expected and once the distinction goes on top of a particular threshold it exits slow begin and enters the congestion avoidance section.

It outline two thresholds value [7] a and b.

- If Diff < a, TCP Vegas increases Congestion Window (CWND) linearly throughout next RTT.
- If Diff > b, TCP Vegas decreases the CWND linearly. If a < Diff < b, TCP Vegas leaves the CWND.

Problems: If sufficient buffer exist in routers that specify which congestion rejection algorithm of TCP Vegas will perform bigger throughput and result of faster reply time. As burden will increase or the quantity of router buffer decreases, congestion rejection algorithms of TCP Vegas is not as in result and begin to act like Reno. In use of router buffer TCP Vegas are fewer violent than Reno as a result of TCP Vegas is restricted. Finally the congestion detection mechanism of TCP Vegas rest on the proper value for Base RTT.

7. Tcp Westwood

TCP Westwood (TCPW) congestion control algorithm use a bandwidth estimation, it is executed at sender side of a TCP connection. The congestion window dynamics throughout slow start and congestion avoidance are unchanged. The general idea is to use the bandwidth estimate BWE to line the congestion window (cwin) and also the slow start threshold (ssthresh) once a congestion episode.

In TCP Westwood the sender continuously computes the connection BWE that is outlined because the share bottleneck used by the connection. Thus, BWE is adequate to the speed at that knowledge is delivered to the transmission control protocol receiver. The estimate relies on the speed at that ACKs are received and on their payload. Once a packet loss, the sender resets the congestion window and also the slow start. Threshold based on BWE. The packet loss is suspected with a reception of three duplicates ACKs or timeout expiration. Another vital part of this procedure is that the RTT estimation. That is as a result congestion window is about exactly to BWE * RTT once the indication of packet loss[6].

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

8. TCP Lite

TCP Lite is a service that gives a transport channel that intercepts communications protocol so as to cut back the overhead in session management within which no application data is transmitted or received. It reduces or eliminates pure TCP protocol data units that are utilized in the setup, teardown and acknowledgment of a channel whereas maintaining order, integrity, reliability and security of the original TCP transport. It is just like to TCP Reno. It detects and re-transmits over one lost packet before timeout occurs. It has higher congestion avoidance and bandwidth utilization over Tahoe and Reno as a result of TCP Lite provides huge window and protection against wrapped sequence numbers choice. It suggests higher method for fast retransmission once packet losses in network. The time in seconds that the TCP Lite transport waits before it retransmits an data segment whose receipt was not acknowledged.

Problems: Once it comes to congestion control, TCP Lite doesn't have several advantages over Reno. However once window will increases it have some issues to keep up them. It will suffer from performance drawbacks since packets are dropped in large amount. It does not reduce the congestion window like that of TCP Reno for congestion avoidance.

Table -1: Comparison study of TCP Variants

Algorithm/Tcp Variants	Tcp Tahoe	Tcp Reno	Tcp New Reno	Tcp SACK	Tcp FACK	Tcp Vegas	Tcp Westwood	Tcp Lite
Slow start	Yes	Yes	Yes	Yes	UV	UV	Yes	Yes
Congestion Control	Yes	Yes	Yes	Yes	Yes	UV	Yes	No
Fast Retransmit	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Fast Recovery	No	Yes	UV	UV	UV	Yes	UV	Yes
Retransmission Mechanism	N	N	N	N	N	NM	N	N
Congestion Control Mechanism	N	N	N	N	NM	NM	N	N
Selective ACK Mechanism	No	No	No	Yes	Yes	No	Yes	No

UV-Updated Version N-Normal NM-New Mechanism

2. Routing Protocols

A routing protocol specifies how routers communicate with each other, distributing information that enables them to select routes between any two nodes on a computer network. It is the act of transferring information (packets) across a network from a source to a destination. Each node maintains a list of all destinations and number of hops to each destination. The sequence number assigned by the destination node. It uses full dump or incremental update to reduce network traffic generated by route updates. Topology-based routing protocols use the information regarding the links that exist within the network to perform packet forwarding. Routing protocols for MANET s can be broadly classified into three categories. Those are Proactive (Table driven) and Reactive (On Demand) and Hybrid routing protocols. Proactive Protocols have lower latency due to maintenance of routes at all times, it can result in much higher overhead due to frequent route updates. When a packet needs to be forwarded, the route is already known. Reactive Protocols may have higher latency since the routes have to be discovered when the source node initiates a route request, lower overhead since routes are maintained only on-demand basis. Hybrid routing protocols merges the best features of table driven and on demand routing protocols.

1. DSDV

Destination-Sequenced Distance-Vector (DSDV) is a table-driven routing scheme for MANETs. This routing protocol is an enhanced version of the distributed Bellman-Ford algorithm where each node maintains a table that contains the shortest distance and the first node on the shortest path to every other node in the network. The sequence number assigned by the destination node. It uses full dump or progressive update to scale back network traffic generated by route updates. The broadcast of route updates is delayed by settling time. Table-Driven protocols like DSDV which maintain a route for each node in its table.

2. OLSR

The Optimized Link State Routing (OLSR) is a table-driven, proactive routing protocol developed for MANETs. It is associate optimization of pure link state protocols that reduces the dimensions of control packets additionally because the range of control packet transmissions needed. OLSR reduces the control traffic overhead by mistreatment Multipoint Relays (MPR), that is that the key plan behind OLSR. An MPR is a nodes one-hop neighbor that has been chosen to forward packets. Rather than pure flooding of the network, packets are forwarded by nodes MPRs. This delimits the network overhead, therefore being a lot of economical than pure link state routing protocols. OLSR is well matched to massive and dense mobile networks. Because of the use of MPRs, the larger and a lot of dense a network, the more optimized link state routing is achieved. MPRs help providing the shortest path to a destination

3. AODV

The Ad Hoc On-Demand Distance Vector routing protocol (AODV) is enhancement of the DSDV protocol. AODV needs nodes to keep up solely active routes. The route discovery process involves RouteRequest (RREQ) and RouteReply (RREP) packets. The source node initiates the route requested through the route discovery process using RREQ packets. The generated route request is forwarded to the neighbors of the source node and this process is repeated till it reaches the destination. The RouteRequest packets produce temporary route entries for the reverse path and is designed to work well even with very high rates of mobility.

4. TORA

The Temporally Ordered Routing Algorithm is an algorithm for routing data across Wireless Mesh Networks or Mobile ad hoc networks. TORA is an on-demand routing protocol. The main objective of TORA is to limit control message propagation in the highly dynamic mobile computing environment. Each node has to explicitly initiate a query when it needs to send data to a particular destination. through each node it passes within the network. When it reaches the destination a RouteReply is distributed back through the identical path the RouteRequest was transmitted. Each node maintains a route table entry that updates the route ending time. A route is valid for the given ending time, when that route entry is deleted from the routing table. Whenever a route is employed to forward the information packet the route ending time is updated to the present time plus the Active Route Timeout. An active neighbor node list is employed by AODV at every node as a route entry to stay keep of the neighboring nodes that are using the entry to route data packets. These nodes are notified with RouteError packets once the link to the following hop node is broken. Every such neighbor node that successively, forwards the Route Error to its own list of active neighbors, so disconfirming all the routes using the broken link. On-Demand protocol finds the route to the destination node by broadcasting route request (RREQ) packets within the entire network once the route is required.

5. DSR

Dynamic Source Routing (DSR) is a simple and economical routing protocol designed specifically to be used in multi-hop wireless ad hoc networks of mobile nodes. DSR permits the network to be utterly self-organizing and self-configuring, while not the requirement for any existing network infrastructure or administration. The protocol consists of the two main mechanisms: · Route Discovery · Route Maintenance that work along to permit nodes to get and maintain routes to arbitrary destinations within the ad hoc network. All aspects of the protocol operate entirely on demand, permitting the routing packet overhead of DSR to scale mechanically to solely what is required to react to changes within the routes presently in use. The protocol permits multiple routes to any destination and allows each sender to select and control the routes employed in routing its packets, as an example, to be used in load balancing or for increased robustness. Other advantages of the DSR protocol include easily guaranteed loop free routing, operation in networks containing one-way links, use of only "soft state" in routing, and very rapid recovery once routes within the network change. The DSR protocol is intended primarily for mobile ad hoc networks of up to about two hundred nodes

TORA essentially performs three tasks:

- Creation of a route from a source to a destination.
- Maintenance of the route.
- Erasure of the route when the route is no longer valid.

3. Conclusion

This paper presents a study of Eight TCP protocols, TCP Tahoe, TCP Reno, TCP New Reno, TCP SACK, TCP FACK, TCP Vegas, TCP Westwood and TCP Lite. This paper concluded that congestion is that the main drawback in different

variants of TCP. Every variant has totally different mechanism to manage congestion in network. However TCP Vegas has improved mechanism to recover the loss of packet due to congestion and corruption in network. It has improved mechanism for Slow Start and Congestion Avoidance and also included new mechanism for Retransmission and Congestion Control. Therefore it will be analysed that TCP Vegas performs better than other TCP variants. An oversized range of various styles of routing protocols are practiced in Mobile Adhoc networks. The result obtained from simulation is DSR performs better than all other routing protocols. Thus reactive routing protocol performs better than proactive routing protocols as regards to Packet delivery ratio and energy consumption.

Reference

1) ArchitMathuNortheastern University, MA, 02115

2) K. Fall, S. Floyd "Simulation Based Comparison of Tahoe, Reno and SACK TCP", 1998.

3) Renaud Bruyeron, Bruno Hemon, Lixia Zhang: "Experimentations with TCP Selective Acknowledgment", ACM SIGCOMM Computer Communication Review, April 1988

4) 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.

5) Abhishek Sawarkar, Saraswat Northeastern University, MA – 02115, Himanshu PES MCOE,Pune-411005:" Performance Analysis of TCP Variants "

6) "Performance Analysis of Different Routing Protocol with Different TCP Variants over Mobile Ad-hoc Network Using NS-2" Urvashi Bharti, Assistant Professor Er. Anu Department of Computer Science & Engineering, SKIET, Haryana, India

7) Pooja Chaudhary, Sachin Kumar, PhD," A Review of Comparative Analysis of TCP Variants for Congestion Control in Network", Department of Computer Science& Engineering Ajay Kumar Garg Engineering College, Ghaziabad, Uttar Pradesh, India

8) K. Fall and S. Floyd. Simulation-based comparisons of Tahoe, Reno, and SACK TCP. In ACM Computer Communication Review, volume 26, pages 5-21, July 1996. ftp:/Iftp.ee.lbl.gov/papersJsacks.ps.Z.

9) Sonia Fahmy and Tapan Prem Karwa,"TCP Congestion Control: Overview and Survey of Ongoing Research".

10) Poonam Tomar, Prashant Panse, "A Comprehensive Analysis and Comparison of TCP Tahoe, TCP Reno and TCP Lite".