ANALYSIS OF DIFFERENT TCP VARIANTS OVER MANET'S USING NS2

Hamid Farooq, Qazi Mohammad Nasir, Kamal Chhabra

Abstract— **A Mobile Ad hoc NETwork (MANET) is one that comes together as needed, not necessarily with any support from the existing Internet infrastructure or any other kind of fixed stations. We can formalize this statement by defining an ad hoc network as an autonomous system of mobile hosts (also serving as routers) connected by wireless links, the union of which forms a communication network modeled in the form of an arbitrary graph. But in wireless networks suffers from significant throughput degradation and delays.**

In this paper we have analyzed the performance of tcp algorithms with AODV, DSR and TORA for throughput. The effect of throughput on the TCP variants New Reno, Reno and Tahoe with different node scenarios was studied.

I. INTRODUCTION

Mobile Ad Hoc Networks (MANETs) are wireless mobile nodes or an autonomous group of mobile users that cooperatively form a network without infrastructure.

There is a direct communication among neighboring devices in MANETs but communication between non-neighboring devices requires a routing algorithm.

A lot of work has been done on routing protocols since they are critical to the functioning of ad-hoc networks [1], [2], [3] Within the two categories of routing protocols described in literature: Proactive and Reactive, it is more suited for highly mobile ad hoc networks due to its ability to cope with rapidly changing network topologies.

Transport Control Protocol /Internet Protocol (TCP/IP) is a connection oriented protocol of the transport layer. It provides features like flow control, reliability and congestion control. It has been very effective in data transmission delivery and have also developed variants to possess the possibility to increase performance and multiple packet loss recovery.

In order to adapt TCP to wireless network, improvements have been proposed in the literature to help TCP to differentiate between the different types of losses. Indeed, in mobile adhoc networks losses are not always due to network congestion, as it is in the wired networks

This protocol is a standard networking protocol on the internet and is the most widely used transport protocol for data services like file transfer, e-mail and WWW browser.

In addition, various routing protocols behave differently over the variants of TCP. It is essential to understand the performance of different MANET routing protocols under TCP variants. In this paper, we have done a performance analysis of MANET Routing Protocols over different TCP Variants.

The paper is organised as follows. Section 1 provides Introduction. Section 2 describes the Standard Routing. Section 3 describes the various TCP variants. Section 4 presents the simulation setup of our work. Finally Section 5 gives the future scope of our work concludes the paper.

II. Routing in AD-Hoc Networks

This network is used to connect the mobile nodes via wireless links. This type of network is known as MOBILE ADHOC NETWORK (MANET). Each device in a MANET is self independent, self organised and infrastructure less network with no fixed BS. The immediate mobile nodes between two nodes act as router to deliver the packets between them. So, MANET is a highly dynamic network.

The routing protocol should posses the following properties, though all of these might not be possible to incorporate in a single solution.

- The routing protocol should be power-efficient.
- A routing protocol should be distributed to increase its reliability.
- Security should be the main factor of any routing protocol.

Manuscript received July 19, 2014

Hamid Farooq , M. Tech. Computer Science Department, GITM, Bilaspur, Haryana, India

Qazi Mohammad Nasir, M. Tech. Computer Science Department, GITM, Bilaspur, Haryana, India

Kamal Chhabra, Assistant Professor, Computer Science Department, GITM, Bilaspur, Haryana, India

- A routing protocol should have good Quality of Service.
- A hybrid routing protocol should be much more reactive than table driven to avoid overheads in the network.
- The different types of communications used in mobile ad hoc networks are:

Unicasting: Unicast is a type of transmission in which only two nodes are exchanging the information. It is a one-to-one communication.

Broadcasting: In Broadcast transmission, information is sent from just one node and received by all the other nodes connected to the network. One to all communication is called as broadcast.

Multicasting: In this type of transmission, the information is sent to a set of nodes. It is a limited case of broadcasting. Multicasting lowers the communication cost for applications that transmits the same data to more recipients.

2.1 Types of routing protocols:

1. Table Driven Routing Protocols (Proactive)

In proactive or table-driven routing protocols, each node maintains a routing table and continuously maintains up-to-date routes to every other node in the network by exchanging the information among all the nodes in the network. So, if there is a need for a route to destination, these routing tables are available to tell the route to destination immediately. As these tables need to maintain node entries for each and every node in the routing table and due to which proactive protocols are not suitable for larger networks. If the network topology changes too frequently, the maintenance the network will be very high.

2. On-Demand routing Protocols (Reactive)

With on-demand routing protocols, if a source Node requires a route to the destination and it does not have the route to destination, so it initiates a route discovery process which goes from one node to the other until it reaches to the destination or an intermediate node that has a route to the destination. This protocol does not need periodic exchange of routing information.

3. Hybrid Routing Protocols

Since, reactive or proactive feature of a particular routing protocol is not enough; so we introduce a better routing protocol which is a mixture of both reactive routing protocol and proactive routing protocol that yield a better solution. Hence, in the recent years, several hybrid protocols are also proposed.

3. TCP VARIANTS

TCP assumes that all the packet loss in the network is due to network congestion, but in mobile adhoc networks losses are not always due to network congestion. Now a day's nearly all the TCP versions assume that packet losses are due to congestion. So when a packet is detected to be lost in wired network, either by Timeout or by multiple duplicate ACKs, TCP slows down the sending rate by adjusting the congestion window. But Wireless networks suffer from several types of losses that are not related to congestion only, making TCP not adapted to this environment. A lot of optimizations and improvements have been proposed to improve TCP performance over wireless networks. There are following TCP versions:

3.1. TCP New-Reno

The New-Reno TCP is a modification of TCP Reno. It improves the retransmissions during the fast recovery phase. In this phase, a new unsent packet is sent from the end of the congestion window to keep the transmit window full, for every duplicate ACK that is returned is also from the end of window, For every ACK, the sender assumes that the ACK points to a new hole, and the next packet beyond the acknowledged sequence number is sent. This progress in the transmit buffer resets the timeout timer, and allows New-Reno to fill large or multiple holes in the sequence space. This process will maintain high throughput, because New-Reno can send new packets at the end of the congestion window during fast recovery.

When enters fast recovery, TCP records the highest sequence number of unacknowledged packet. Upon the acknowledgment of this sequence number, TCP returns to the congestion avoidance state. New-Reno will misinterpret the situation if there are no losses, but instead reordering of packets by more than 3 packet sequence numbers. In such a case, New-Reno mistakenly enters fast recovery, but when there is any reordered packet is delivered, Acknowledgement sequence-number increments occurs and from there until the end of fast recovery, every bit of sequence number increments produces a duplicate and needless retransmission that is immediately acknowledged.

3.2. TCP Vegas

It provides a TCP congestion avoidance algorithm that uses packet delay rather than packet loss. The Vegas congestion detection algorithm differs from earlier TCP Tahoe and Reno where congestion is detected by packet drops only after it has actually happened. TCP Vegas detects congestion at a stage based on increasing RTT values of the packets in the connection. TCP Vegas can identify the queuing delay and based on this, it will adjust the congestion window size. The difference between expected traffic and actual traffic is used to adjust the size of the congestion

window. Both the increase and decrease of the rate is additive (Additive Increase, Additive Decrease (AIAD)). But in TCP Reno, Congestion window keep increasing until a packet is lost, and therefore they will always face packet loss at some point or other. Vegas give 40 to 70% better throughput than TCP Reno with less than half the packet loss. In addition to the modified congestion avoidance mechanism, the TCP Vegas also uses the retransmission mechanism to avoid timeout. If the sender is unable to receive 3 duplicate ACKs (due to lose segments or window size is too small), in such a case, the sender can do retransmission after one dup ACK is received, if RTT estimate > timeout.
The slow start phase is modified so that the sender tries to find the correct window size without causing a loss. The Vegas algorithm heavily depends on accurate calculation of the base RTT value. If it is too small, then the throughput of the connection will be less, while if the value is too large, it will push too much traffic over the network path.

3.3 TCP SACK

The client sends request to the server, and the server gives a response that is broken into four TCP segments. The server transmits all four packets in response to the request but the second response packet is dropped somewhere in the network and never reaches the host. Let's discuss what happens.

Step 1
Data segment 2 is lost, while responding.

Step 2
The client receives segment 3. The client realizes this segment is out of order by examining the segment's sequence. It means there is data missing between the last segment received and this one. The client transmits a duplicate acknowledgment for packet 1 to alert the server that it has not received any data beyond packet 1.

Step 3
The server is not yet aware that anything is wrong because it has not yet received the client's duplicate acknowledgment, it continues to send segment 4. The client then realizes that data packet 2 is still missing, and repeats its behaviour in step three by sending another duplicate acknowledgment for packet 1.

Step 4
The server receives the client's first duplicate acknowledgment for packet 1. Because the client has only received the first segment out of the four segments, so the server must retransmit all three remaining segments in the response. The second duplicate acknowledgment received from the client is ignored.

Step 5
The client successfully acknowledges for three remaining segments that are received now.

Enter Selective Acknowledgments
We've noticed that this design is inefficient: as only packet 2 was missed, the server was required to retransmit packets 3 and 4 as well, because the client had no way to confirm that it had received those packets. This problem can be solved by introducing the selective acknowledgment (SACK) TCP option. SACK works as follows:
It allows the client to say "I have not received data 2, but I have received data segment 3 and 4"

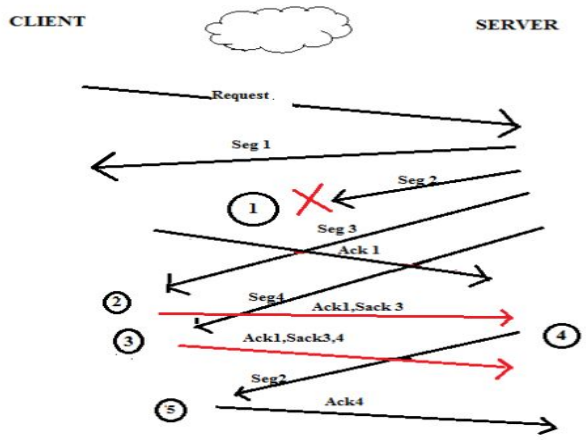

Figure 4 TCP sack

Step 1
Response segment 2 is lost.

Step 2
The client realizes it is missing a segment between segments 1 and 3. It sends a duplicate acknowledgment for segment 1, and also uses a SACK option indicating that it has received segment 3.

Step 3
The client receives data segment 4 and then sends another duplicate acknowledgment for segment 1, but this time SACK option used to show that it has received segments 3 through 4.

Step 4
The server receives the client's duplicate ACK for segment 1 and SACK for segment 3. By this, the server can understand that the client has not received data segment 2, so segment 2 is retransmitted again. The next Selective Acknowledgement (SACK) received by the server indicates that the client has received segment 4 successfully.

Step 5
The client receives segment 2 and then sends an acknowledgment to indicate that client has received all data up to an including segment 4.

3.4 TCP WESTWOOD

Westwood TCP is a new congestion control algorithm that is based on end-to-end bandwidth estimate. In particular, TCP Westwood estimates the available bandwidth by counting and filtering the flow of returning ACKs and adaptively sets the *cwnd* and the *ssthresh* after congestion by taking into account the estimated bandwidth. The original bandwidth estimation algorithm fails to work properly in the presence of ACK compression. Thus a slightly modified version of the bandwidth estimation algorithm has been proposed in [14] to cope with ACK compression effect. We call Westwood the original Westwood algorithm with the enhanced bandwidth estimate. Furthermore, in [14] it has been shown via a mathematical analysis that Westwood is friendly towards Reno TCP and fairer than Reno in bandwidth allocation.

The Westwood algorithm is based on end-to-end estimation of the bandwidth available along the TCP connection path [13],[14]. The estimate is obtained by filtering the stream of returning ACK packets and it is used to adaptively set the control windows when network congestion is experienced. In particular, when three DUPACKs are received, both the congestion window (*cwnd*) and the slow start threshold (*ssthresh*) are set equal to the estimated bandwidth (*BWE*) times the minimum measured round trip time (*RTTmin*); when a coarse timeout expires the *ssthresh* is set as before while the *cwnd* is set equal to one.

The pseudo code of the Westwood algorithm is reported below:

a) On ACK reception:

cwnd is increased accordingly to the Reno algorithm;
the end-to-end bandwidth estimate *BWE* is computed;

b) When 3 DUPACKs are received:

ssthresh =max(2, (*BWE* RTTmin*) / *seg_size*);
cwnd = *ssthresh*;

c) When coarse timeout expires:
ssthresh = max(2,(*BWE* RTTmin*) / *seg_size*);
cwnd = 1;

From the pseudo-code reported above, it turns out that Westwood additively increases the *cwnd* as Reno, when ACKs are received. On the other hand, when a congestion episode happens, Westwood employs an adaptive setting of *cwnd* and *ssthresh* so that it can be said that Westwood+ follows an *Additive-Increase/Adaptive-Decrease* paradigm [14]. It is worth noting that the adaptive decrease mechanism employed by Westwood TCP improves the stability of the standard TCP multiplicative decrease algorithm. In fact, the adaptive window shrinking provides a congestion window that is decreased enough in the presence of heavy congestion and not too much in the presence of light congestion or losses that are not due to congestion, such as in the case of unreliable radio links. Moreover, the adaptive setting of the control windows increases the fair allocation of available bandwidth to different TCP flows. This result can be intuitively explained by considering that the window setting of Westwood TCP tracks the estimated bandwidth so that, if this estimate is a good measurement of the fair share, then the fairness is improved. Alternatively, it could be noted that the setting *cwnd* = *B.RTTmin* sustains a transmission rate (*cwnd/RTT*) = (*B.RTTmin*)/*RTT* that is smaller than the bandwidth *B* estimated at the time of congestion: as a consequence, the Westwood TCP flow clears out its path backlog after the setting thus leaving room in the buffers for coexisting flows, which improves statistical multiplexing and fairness.

III. SIMULATION

We have evaluated the performance of different variants of TCP using OPNET simulator. Here AODV, DSR and TORA are simulated with different TCP algorithms with scenario of 50 nodes.

4.1 TCP in AODV

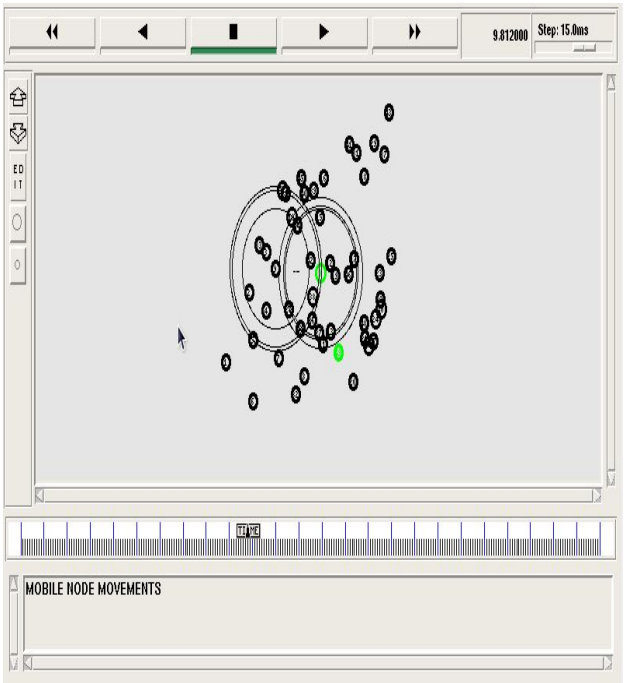

Figure5.2: TCP Simulation with 50 nodes in AODV.

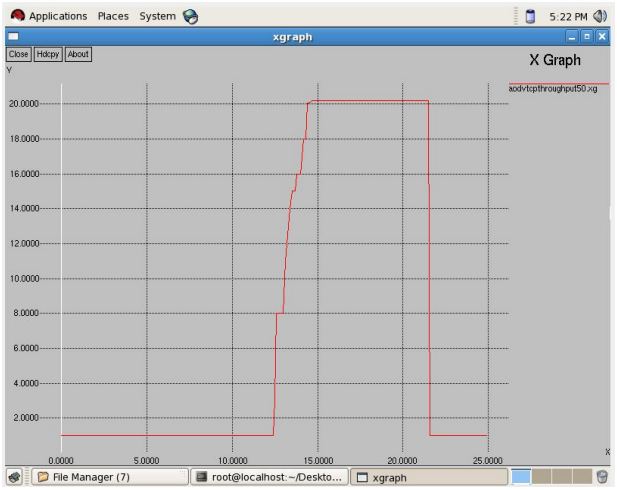

Figure5.3: X-Graph representing throughput for TCP Simulation

5.3.2 TCP New Reno in AODV

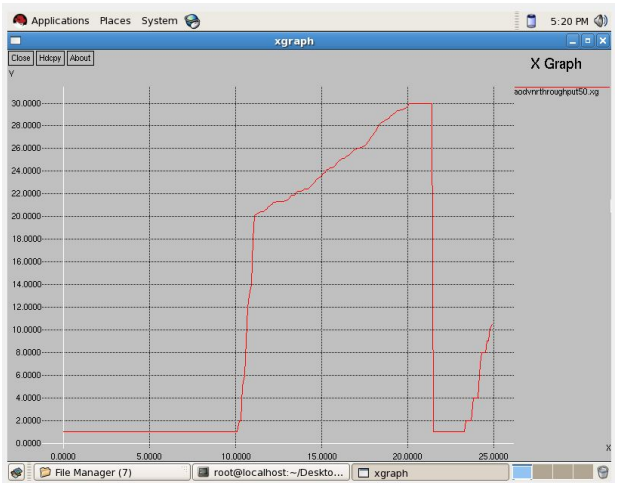

Figure 5.4: X-graph representing throughput for TCP NEW-RENO Simulation

5.3.3 TCP Vegas in AODV

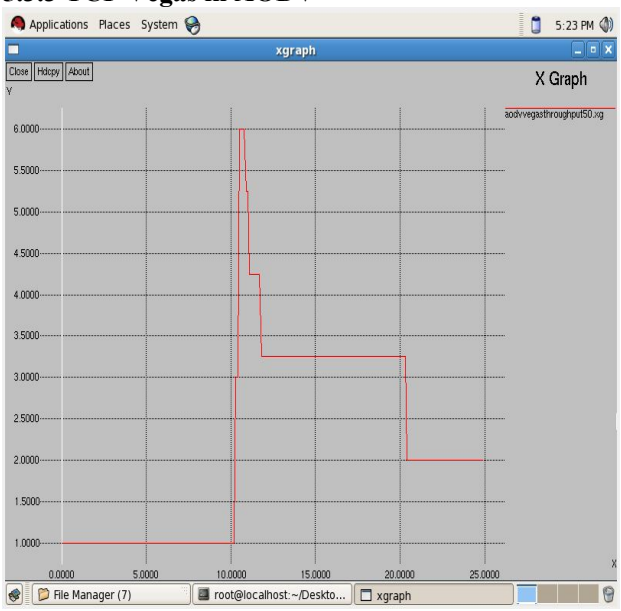

Figure 5.5: X-graph representing throughput for TCP VEGAS Simulation

5.3.4 TCP Sack in AODV

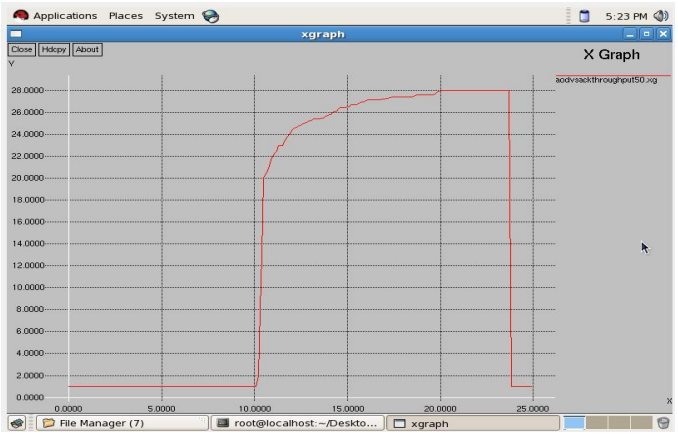

Figure 5.6: X-graph representing throughput for TCP SACK Simulation

5.3.5 TCP WESTWOOD in AODV

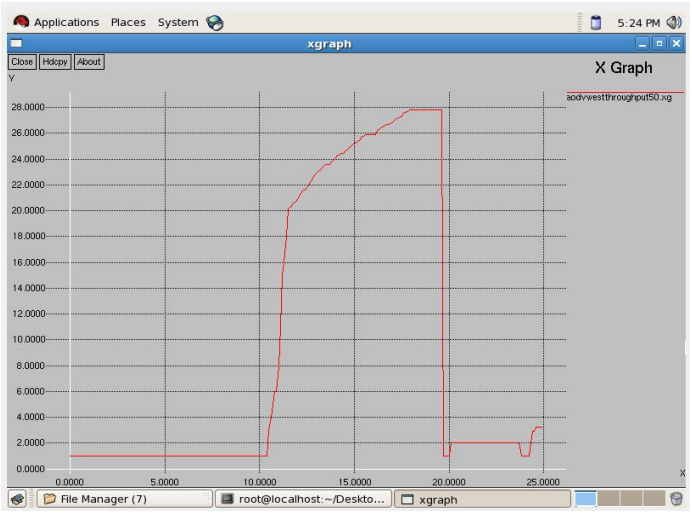

Figure5.7: X-Graph representing throughput for TCP Westwood Simulation

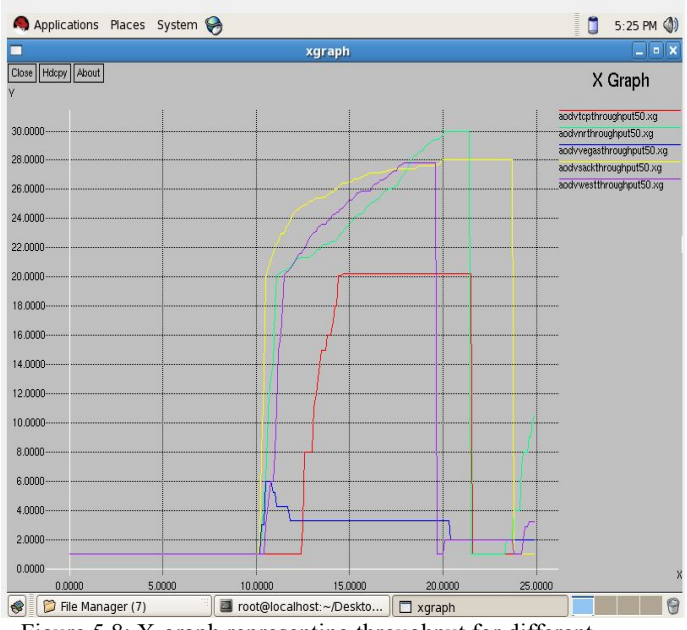

Figure 5.8: X-graph representing throughput for different TCP variants.

In figure 5.8, throughput comparison is shown. Here we can clearly see that tcp-vegas perform worst in AODV (Adhoc On-demand Distance Vector) protocol. And tcp-newreno perform best in this case.

5.3.6 TCP in DSDV

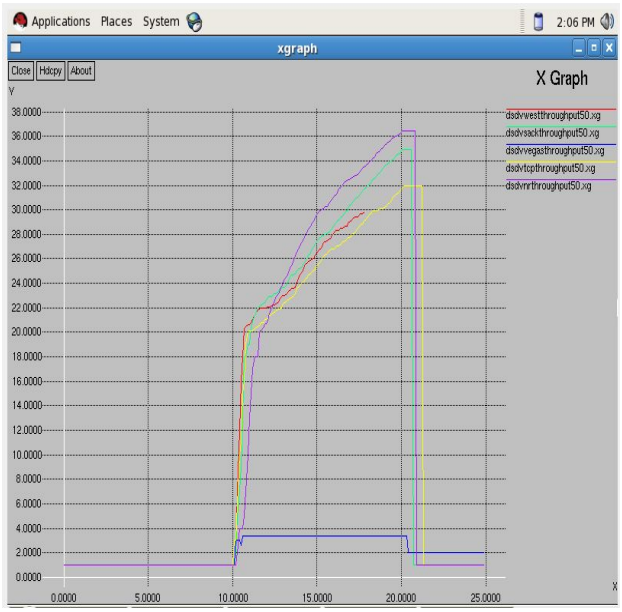

Figure 5.15: X-graph representing throughput for different TCP variants in DSDV.

In figure 5.15, throughput comparison is shown. Here we can clearly see that tcp-vegas perform worst in DSDV (Destination Sequence Distance Vector) protocol. And tcp-newreno perform best in this case.

5.3.11 TCP in DSR

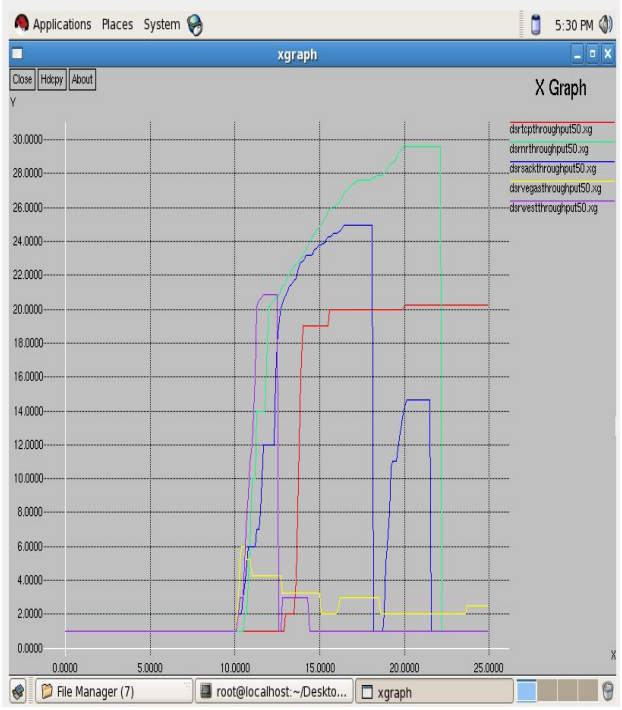

Figure 5.22: X-graph representing throughput for different TCP variants in DSR.

In figure 5.22, throughput comparison is shown. Here we can clearly see that tcp-vegas perform worst in DSR (Destination Sequence Routing) protocol. And tcp-newreno perform best in this case.

CONCLUSION

In this work we perform a simulation analysis of different TCP varients over different routing protocols of Mobile Adhoc Networks. In the proposed work approach,we presented the simulation of TCP, TCP New reno, Vegas, Sack and Westwood over AODV, DSDV and DSR in MANET. The Simulation shows the 50 nodes and data transfer between these nodes. It also shows the X-graphs of throughput in simulation. Throughput is based on packet delivery between 50 nodes. The implementation is performed in NS2 and analysis is presented using X-graphs.

FUTURE WORK

The Proposed system can be enhanced in future by other researchers by working over other different TCP's with other different protocols.

REFERENCES

1. C.Sivaram Murthy and B.S. Manoj,"ADHOC Wireless Networks: Architecture and Protocols", Prentice Hall PTR, 2004.
2. Dong kyun Kim , Juan-Carlos Cano and P. Manzoni, C-K. Toh," A comparison of the performance of TCP-Reno and TCP-Vegas over MANETs", 1-4244-0398-7/06/$20.00 ©2006 IEEE
3. Foez ahmed, Sateesh Kumar Pradhan, Nayeema Islam, and Sumon Kumar Debnath," Performance Evaluation of TCP over Mobile Ad-hoc Networks" , (IJCSIS) International Journal of Computer Science and Information Security,Vol. 7, No. 1, 2010.
4. Xiang Chen, Hongqiang Zhai, Jianfeng Wang, and Yuguang Fang ,"TCP performance over mobile ad hoc networks",CAN. J. ELECT. COMPUT. ENG., VOL. 29, NO. 1/2, JANUARY/APRIL 2004
5. GAVIN HOLLAND NITIN VAIDYA," Analysis of TCP Performance over Mobile Ad Hoc Networks", Wireless Networks 8, 275–288, 2002 □□2002 Kluwer Academic Publishers. Manufactured in the Netherlands
6. Harpreet Singh Chawla, M. I. H. Ansari, Ashish Kumar, Prashant Singh Yadav," A Survey of TCP over Mobile ADHOC Networks", International Journal of Scientific & Technology Research Volume 1, Issue 4, May 2012 ISSN 2277-8616
7. Bogdan Moraru Flavius Copaciu Gabriel Lazar Virgil Dobrota," Practical Analysis of TCP Implementations: Tahoe, Reno, NewReno"
8. Ashish Ahuja, Sulabh Agarwal, Jatinder Pal Singh, Rajeev Shorey," Performance of TCP over Different Routing Protocols in Mobile Ad-Hoc Networks,"0-7803-571 8-3/00/$10.00 *02000* IEEE.
9. Laxmi Subedi, Mohamadreza Najiminaini, and Ljiljana Trajković," Performance Evaluation of

TCP Tahoe, Reno, Reno with SACK, and NewReno Using OPNET Modeler"

10. Ahmad Al Hanbali, Eitan Altan Altman, and Philippe Nain, Inria Sophia Antipolis France," A SURVEY OF TCP OVER AD HOC NETWORKS ",1553-877X IEEE Communications Surveys & Tutorials • Third Quarter 2005.
11. Chaoyue Xiong, Jaegeol Yim, Jason Leigh and Tadao Murata, "Energy-Efficient Method to Improve TCP Performance for MANETs"
12. Ahmad Al Hanbali, Eitan Altman, Philippe Nain," A Survey of TCP over Ad Hoc Network.
13. Mascolo, S., Casetti, C., Gerla, M., Sanadidi, M., Wang, R. TCP Westwood: End-to-End Bandwidth Estimation for Efficient Transport over Wired and Wireless Networks, in Proceedings of ACM Mobicom 2001, (Rome, Italy, July 2001).
14. Grieco, L. A., and Mascolo, S., Westwood TCP and easy RED to improve Fairness in High Speed Networks, in Proceedings of IFIP/IEEE Seventh International Workshop on Protocols For High-Speed Networks, PfHSN02, (Berlin, Germany, April, 2002).
15. H. Balakrishnan, V.N. Padmanabhan, S. Seshan and R.H. Katz, A comparison of mechanisms for improving TCP performance over wireless links, IEEE/ACM Transactions on Networking (December 1997).
16. H. Balakrishnan, S. Seshan, E. Amir and R.H. Katz, Improving TCP/IP performance over wireless networks, in: *MOBICOM'95*, Berkeley, CA (November 1995).
17. H. Balakrishnan and R.H. Katz, Explicit loss notification and wireless web performance, in: *Proceedings of IEEE GLOBECOM'98 Internet Mini-Conference*, Sydney, Australia (November 1998).
18. T. Bonald, Comparison of TCP Reno and TCP Vegas: Efficiency and fairness, in: *Proceedings of PERFORMANCE'99*, Istanbul, Turkey (October 1999).
19. C. Casetti, M. Gerla, S. Lee, S. Mascolo and M. Sanadidi, TCP with faster recovery, in: *MILCOM 2000*, Los Angeles, CA (October 2000).
20. D. Clark, The design philosophy of the DARPA Internet protocols, in: *Proceedings of SIGCOMM'88*, ACM Computer Communication Review 18(4) (1988) 106–114.