TCP PERFORMANCE ANALYSIS FOR MOBILE AD HOC NETWORK USING ON-DEMAND ROUTING PROTOCOLS

K.Kathiravan
B S Abdur Rehman Crescent Engineering College
Vandalur, Chennai – 48

Dr. S. Thamarai Selvi
Professor
MIT Chromepet Campus
Anna University, Chennai – 14

A.Selvam
BSA Crescent Engineering College
Vandalur, Chennai – 48.

ABSTRACT

TCP optimization in mobile ad hoc networks MANETs is a challenging issue because of some unique characteristics of MANETs. Packet losses in MANETs are mainly due to congestion and frequent link failures but in case of wired networks packet losses are mainly due to congestion. To optimize TCP in MANETs we use congestion control and avoidance algorithms. In this paper, we describe an NS2-based simulation analysis of TCP using omni antennas over mobile ad-hoc network. In particular, we compare the performance of end to end protocols such as TCP-Newreno and TCP-SACK with the routing provided by AODV, DSR and DSDV protocols using omni directional antenna. We investigate the effects of varying node density, mobility of nodes and pause time of nodes has on TCP performance. Through simulation, we show that TCP throughput drops significantly when nodes move, due to TCP's inability to recognize the difference between link failure and congestion. We compare the throughput performance between the TCP versions.

Keywords: TCP, mobile ad hoc networks, wireless networks, Directional antenna.

1 INTRODUCTION

Mobile ad hoc networks recent activities strongly indicate that mobile networks will be an integral part of future internetworks. On the other hand, the performance of the internet protocols in wireless networks has been reported to be much lower than in fixed networks. The main reason for the performance degradation is that the Transmission Control Protocol (TCP) works less efficiently in wireless networks. This problem is important, since TCP/IP is used by many Internet applications, such as e-mail, web browsing and remote login.

Basically, TCP/IP protocol was designed for wired networks which provides end to end reliable communication between nodes and assures ordered delivery of packets. It also provides flow control and error control mechanisms. As it is still a successful protocol in wired networks, we adapt it to mobile ad hoc networks. In wired networks that uses packet losses are mainly due to congestion. But in case of ad hoc networks packet losses are due to congestion in the network and due to frequent link failures so when we adapt TCP to ad hoc networks it misinterprets the packet losses due to link failure as packet losses due to congestion and in the instance of a timeout, backing-off its retransmission timeout (RTO).This results in unnecessary reduction of transmission rate because of which throughput of the whole network degrades.

In the presence of the high error rates and intermittent connectivity characteristics of wireless links, TCP reacts to packet losses as it would in the wired environment so it drops its transmission window size before retransmitting packets, initiates congestion control or avoidance mechanisms such as slow start[9] and resets its retransmission timer [10]. These measures result in an unnecessary reduction in the link's bandwidth utilization, thereby causing a significant degradation in performance in the form of

poor throughput and very high interactive delays [11]. Therefore, route changes due to host mobility can have a detrimental impact on TCP performance.

On-demand Routing Protocols such as AODV and DSR are used for this performance analysis of TCP. These types of routing protocols create routes only when requested by a source node. When a node wants to establish a route to a destination, it initiates a route discovery process within the network. One the route has been established, it is maintained until either destination becomes inaccessible or the route is no longer desired.

In this paper we will analyze the performance of two on-demand routing protocols for ad hoc networks, namely Dynamic Source Routing(DSR) [12] and Ad Hoc On-demand Distance Vector (AODV) [13] routing protocols, based on TCP traffic flows[14-19]. We also use DSDV which is a table driven routing protocol.

The rest of the paper is organized as follows. Section 2 introduces two most popular TCP protocols. Section 3 describes our simulation setup of our work. Section 4 discusses the effect of mobility on TCP performance using different routing protocols. Section 5 discusses about our future framework proposed. Finally Section 6 concludes this paper.

2 RELATED WORKS

TCP optimization in MANETs has been investigated in several studies. TCP does not have any resilience mechanisms that are specially designed to deal with link failures. From the viewpoint of TCP, there is no difference between link failure and network congestion. As a result, when part of the network fails and some segments are dropped, TCP will assume that there is congestion somewhere in the network, and the TCP congestion control mechanisms will start dealing with the segment loss. TCP congestion control mechanisms have improved over time. The main versions of TCP are Tahoe TCP, Reno TCP, NewReno TCP and SACK TCP. Tahoe TCP is the oldest version and only a few old systems use it. Reno TCP, NewReno TCP and SACK TCP are widely implemented [1]. This paper focuses on SACK and NewReno TCP because they are the newer versions and are more widely deployed. Details about TCP congestion control can be found in [2-6]. In our experiments, the TCP implementation conforms to the one illustrated in [5]. Both SACK and NewReno TCP congestion control are composed of three parts: slow start, congestion avoidance and fast retransmit/fast recovery. Three state variables, cwnd (congestion window), rwnd (receiver's advertised window) and ssthresh (slow start threshold), are maintained at the sender to deal with network congestion. In addition, SACK TCP has an extra variable called pipeat the sender that represents the estimated number of outstanding segments. SACK TCP also has a data structure called scoreboard at the sender side that keeps track of the contiguous data blocks that have arrived at the receiver. Retransmission timeout (RTO) is an important parameter in TCP congestion control. It has a minimum of one second and RFC 2988 [8] suggests that a maximum value may be placed on RTO.

Because routing is an important problem in ad hoc network we go for on demand routing protocols such as DSR and AODV using TCP traffic which shows better performance when compared to other routing protocols.

2.1 TCP- NEWRENO

We include New-Reno TCP in this paper to show how a simple change to TCP makes it possible to avoid some of the performance problems of Reno TCP without the addition of SACK. At the same time, we use New-Reno TCP to explore the fundamental limitations of TCP performance in the absence of SACK.

New Reno modifies the Fast Retransmit and Fast Recovery. These modifications are intended to fix the Reno problems above and are wholly implemented in the sender side.A modification of Reno lead to New-Reno TCP which shows that Reno can be improved without the addition of SACKs but still suffers without it. Here, the wait for a retransmit timer is eliminated when multiple packets are lost from a window.

New Reno is the same as Reno but with more intelligence during fast recovery. It utilizes the idea of partial acks: when there are multiple packet drops, the acks for the retransmitted packet will acknowledge some, but not all the segments send before the Fast Retransmit.

- In TCP Reno, the first partial ACK will bring the sender out of the fast recovery phase. This will result in the requirement of timeouts when there are multiple losses in a window, and thus stalling the tcp connection.
- In New Reno, a partial ack is taken as an indication of another lost packet and as such the sender retransmits the first unacknowledged packet. Unlike Reno, partial acks don't take New Reno out of Fast Recovery. This way, it retransmits one packet per RTT until all the lost packets are retransmitted and avoids requiring multiple fast retransmits from a single window of data.

The downside of this is that it may take many RTT's to recover from a loss episode, and you must have enough new data around to keep the ack clock running. This is implemented as follows:

2.2 Multiple packet loss

A fix for Fast Recovery to prevent starting Fast Retransmit and Fast Recovery in succession when multiple segments are dropped in the same window. When entering Fast Retransmit (from 3 dupacks), ave the highest sequence number sent so far. Perform retransmission and the Fast Recovery algorithm as usual (set ssthresh, inflating cwnd on dupacks). When a new ack arrives, perform the addition check if the ack covers the highest sequence number sent when Fast Retransmit was invoked. If not, this ack is a partial ack and signals that another segment was lost from the same window of data. As such, retransmit the segment reported as expected by the partial ack, reset the retransmission timer but do not exit Fast Recovery. On the other hand, if the new ack covers the highest sequence number sent and then exits Fast Recovery but setting cwnd to ssthresh and performing congestion avoidance.

2.3 False Fast Retransmits

Record the highest sequence number ever transmitted after a retransmission timeout (normally set to 0). When ever we get 3 dupacks, we perform a test to see if we should enter Fast Retransmit. If these acks cover the sequence number saved at the previous timeout, then this is a new invocation of the Fast Retransmit. In this case, enter Fast Retransmit and perform the related actions. If they do not cover the sequence numbers (i.e. they ack segments sent previous to the timeout) then just acknowledge the receipt of already queued segments at the receiver. In this case, do not enter Fast Retransmit. Sender comes out of fast recovery only after all outstanding packets (at the time of first loss) are acknowledged.

2.4 TCP-SACK

The selective retransmission strategy [14] is more complex and requires more buffer space at the end-points. Hence TCP traditionally uses cumulative acknowledgements and the of-back-N strategy. Using selective retransmit reduces the overhead of retransmission on errors and therefore cannot be ruled out for use in wireless domains. The TCP with selective acknowledgement scheme (TCP SACK) [4] improves TCP performance by allowing the TCP sender to retransmit packets based on selective ACKs provided by the receiver.

3 PERFORMANCE EVALUATION

We now briefly describe our simulation setup. We use the network simulator ns2 for al our simulations. The setdest tool in ns2 is used to generate the random topologies for the simulations. All simulations are performed for a 1000m × 1000m grid consisting of 50nodes, distributed randomly over the two-dimensional grid. The source-destination pairs are randomly chosen from the set of 50 nodes in the network. We consider seven different speeds of 10m/s, 20m/s, 30m/s, 40m/s in our simulations all with pause time of 0. Two runs were conducted for each of the average speeds and we used, resulting different node density of 20, 40, 60, 80 nodes resulting 128 different movement patterns. Pause time 0 means each node moves constantly throughout the simulation.

TCP packet size of 1460 is considered in our analysis. The TCP clock granularity is set to 200ms. The queue size s are set to 50 packets to avoid frequent drop of packets due to buffering. We measured the TCP throughput for each setup when operating over a wireless system.

3.1 Performance Evaluation

For different node density in a fixed area, we performed a series of three simulation runs. Each simulation run tested a different technique: TCP Newreno and TCP SACK. In each run a set of performance measurements were made for each of the two routing protocols at each of several node densities from 20 nodes to 100 nodes.

Discussion of Figure 1. : We use TCP Newreno for all the measurement of throughput. When 20 nodes are used in the grid, DSR routing protocol performs very well then the other two protocols i.e. AODV and DSDV. AODV and DSDV almost give same throughput.

However, when the nodes increase in the grid throughput also changes. Now when we consider the other set of measurement we notice that the throughput of the network gradually increases on using AODV routing protocol and also for DSDV but for DSR routing protocol the throughput is continuously varying at different node densities and it provides high throughput when we use 60nodes in the grid.

On an average DSR routing protocol provides a better throughput compared to the other routing protocols for this setup.

We use TCP SACK for all the measurement of throughput. It is almost the same case as in TCP Newreno except some measurements. Here we obtain the maximum throughput value on using DSR routing protocol when 60 nodes are used in the grid.

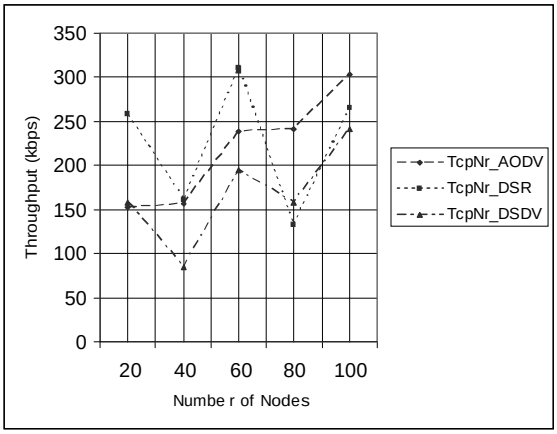

Figure 1. TCP Newreno for various node density and different routing protocols

On an average DSR routing protocol provides a better throughput compared to the other routing protocols for this setup.

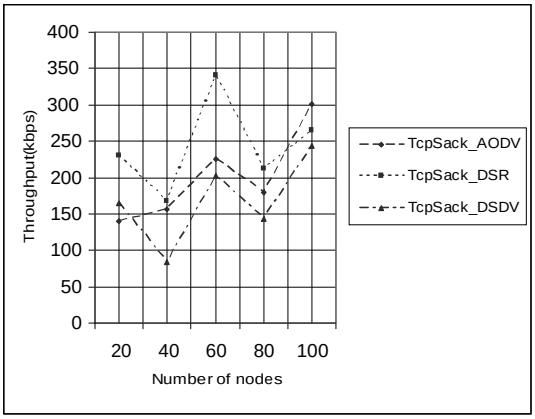

Figure 2. TCP Sack for various node density and different routing protocols

3.1.1 Performance of TCP NEWRENO

We use TCP-Newreno using AODV for different measurements of throughput. These measurement are taken for different mobility conditions from 10 m/s to 40 m/s. The variable quantity kept here is the number of nodes varying from 20 nodes to 80nodes in the grid.

At low speed, we can observe that as the node density increases the throughput decreases. It provides a good throughput when 20 numbers of nodes are used compared to the other throughputs.

But as the speed of nodes varies this result fails. At higher mobility as the node density increases the throughput increases. This is because the nodes at higher mobility go for frequent route failure because of which chances of discovering shortest path between nodes is possible. It is not so when nodes are moving slowly.

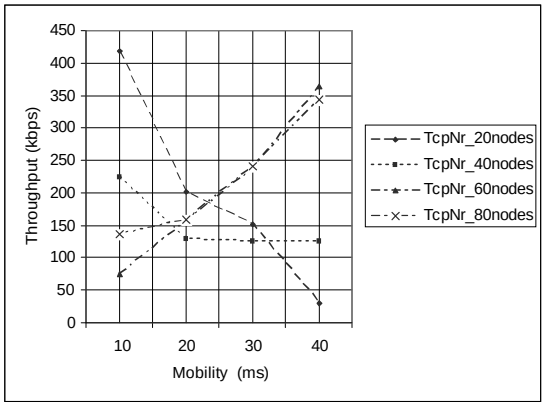

Figure 3. TCP Newreno's performance using AODV for different mobility and different node density

We use TCP-Newreno using DSR for different measurements of throughput. Other setup are same as the previous setup. At low speed, we can observe that as the node density increases the throughput decreases. It provides a good throughput when 20 numbers of nodes are used compared to the other throughputs.

But as the speed of nodes varies this result fails. At higher mobility as the node density increases the throughput increases. This is because the nodes at higher mobility go for frequent route failure because of which chances of discovering shortest path between nodes is possible. It is not so when nodes are moving slowly.

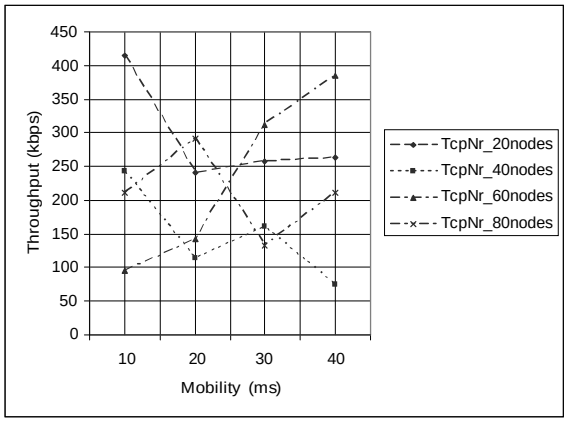

Figure 4. TCP Newreno's performance using DSR for different mobility and different node density

3.1.2 Performance of TCP SACK

We use TCP-SACK using AODV for different measurements of throughput. Other setup

are same as the previous setup.

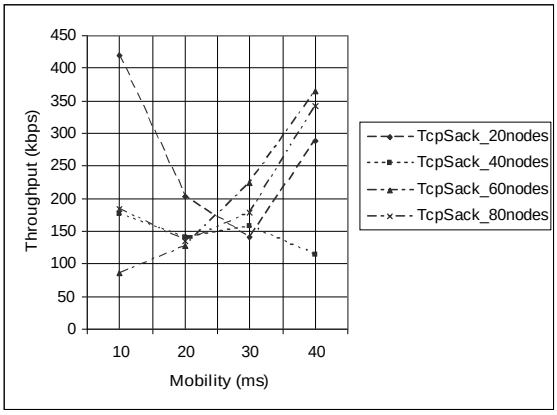

Figure 5. TCP Sack's performance using AODV for different mobility and different node density

We use TCP-SACK using DSR for different measurements of throughput. Other setup are same as the previous setup.

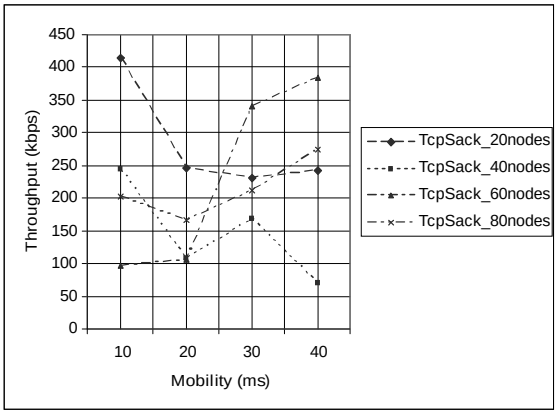

Figure 6. TCP Sack's performance using DSR for different mobility and different node density

On an average TCP SACK performs better than TCP Newreno protocol for different mobility conditions using on-demand routing protocols. Allow two blank lines between the title and the author's full name. Author names are set in Times New Roman type face with 10 point in size, bold and centered. Authors from the same affiliations should be grouped.

4 PROPOSED FRAMEWORK FOR TCP PERFORMANCE IMPROVEMENT USING DIRECTIONAL ANTENNAS

We used NS2 simulator to implement a simulation based analysis of TCP using directional antennas over mobile ad-hoc network. In particular, we aim to show how TCP's performance using omni-directional antenna gets affected by mobility and lower layers protocols. We will investigate the effect of link breakage due to mobility has on TCP performance. Through simulation, we try to show that TCP throughput drops significantly when nodes move away from each other, due to TCP's inability to recognize the difference between link failure and congestion and we aim to overcome this problem by using directional antenna.

An example of a network with 6 nodes is shown in the Figure 7, where nodes in Figure 7(a), uses omni-directional antenna and nodes in Figure 7(b), uses directional antenna. In Figure 7(a), node E is currently communicating with nodes F and G in which the shaded portion represents the transmission range of node E. But because of hidden terminal problem (between node A and node C) and exposed terminal problems (between node C and node D) node A cannot communicate with node D when node E is communicating. But this problem can be overcome with the help directional antenna.

In Figure 7(b), node E is communicating with nodes G and F without interrupting node A's transmission. Thus directional antenna resolves both hidden and exposed problems. It also improves scalability of the network as more number of nodes can communicate simultaneously increasing the network throughput. Wreless communication systems that employ directional antennas for both transmission and reception can provide spatial reuse, bandwidth conservation, increased capacity and range, reduced interference, supports new services such as location estimation, better security, and reduced multipath propagation. For this reason, directional antennas are considered as a key ingredient in the ad-hoc networks, while reliable data services for TCP/IP applications is provided by TCP protocol.By using directional antenna frequent link failure avoided because the transmission range of directional antenna is more than the transmission range of omni-directional antenna. So unnecessary packet losses can be avoided which increases the network throughput.

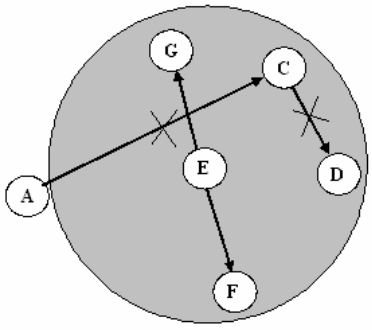

Figure 7. (a) Nodes using omni-directional antenna

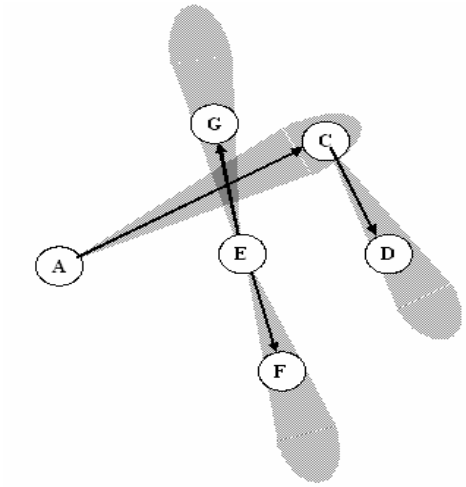

Figure 7. (b) Nodes using directional antenna.

4.1 Capacity of chain nodes

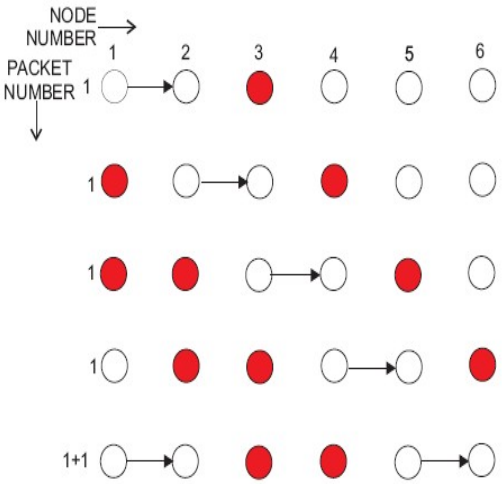

Figure 8. Flow diagrams for chain capacity calculation omni directional antenna

n this case consider the structure shown in Figures 8 and 9. In theory, the entire network in which all the Horizontal flows operate in one time cycle to calculate chain capacity is assumed. For the given link capacity of the channel, larger the number of packets larger the transfer time. The linear chain model was used for calculating the channel capacity with omni directional and directional communication.

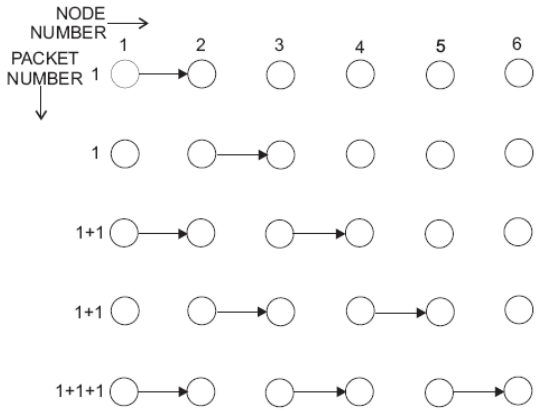

Figure 9. Flow diagrams for chain capacity calculation directional antenna

The general expression relating packets, time-delay and number of nodes were obtained as follows.

4.1.1 For omni-directional case

Let P be the number of packets to be transmitted, T be the time delay, N be the node, then

For $N = 1$ to 4

$$T = (N - 1) \times P \quad \text{units}$$

For $N \geq 5$

$$T = 4 \times [(N - 1/4) + (P - 1)] + [(N - 1) \bmod 4]$$

4.1.2 For directional case

For $N = 1$ to 2 $\quad T = (N - 1) \times P$

For $N \geq 3$

$$T = 2 \times [(N - 1/2) + (P - 1)] + [(N - 1) \bmod 2]$$

Figures 10 and 11 are plotted according to following assumptions. The nodes have an interference range or receiving range of 550 m and transmitting range of 250m.The nodes are communicating as though they are connected together by a 2Mbps full duplex line.

The graphs show the transfer time versus number of nodes in the chain. The plot is given for various numbers of data packets to be carried for each case. The Directional antenna incorporated nodes performs better when compared with Omni directional antenna nodes.

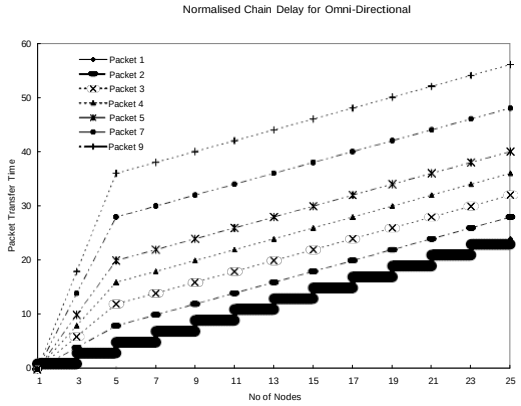

Figure 10. The chain capacity plot for omni directional antenna

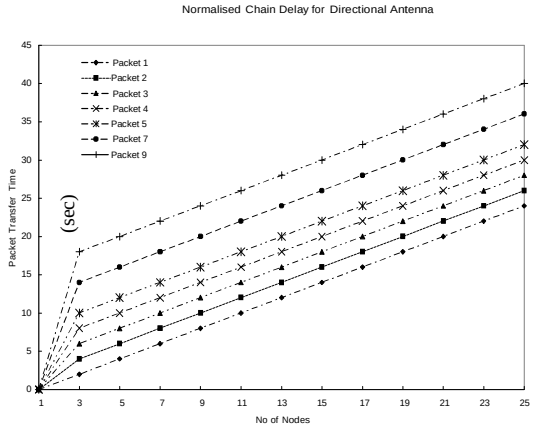

Figure 11. The chain capacity plot for directional antenna

5 CONCLUSION

This paper has presented an overview of TCP Newreno and Sack behavior in a simple mobility scenario under two popular routing protocols in MANETs, namely AODV and DSR. The subsequent study of traces has revealed that TCP Sack performance is slightly better than TCP Newreno and DSR interacts with TCP more efficiently than the other protocol under different mobility conditions as it salvages packets and restores the route quickly in the examined topology. AODV was shown to behave competently by avoiding RTOs through caching at the source of outgoing TCP packets when it has been informed of that a route breakage has occurred.

Through simulation, we shown that TCP throughput drops significantly when nodes move away from each other, due to TCP's inability to recognize the difference between link failure and congestion and we can overcome this problem by using directional antenna.

REFERENCES

[1] J. Pahdye and S. Floyd. On Inferring TCP Behavior. Proceedings of the 2001 conference on applications, technologies, architectures and protocols for computer communications.

[2] M. Allman and V. Paxson. RFC 2581: TCP Congestion Control. Apr. 1999.

[3] W.R. Stevens. TCP/IP Illustrated, Volume 1. Addison Wesley Press , Apr. 2000.

[4] M. Mathis, J. Mahdavi et al. RFC 2018: TCP Selective Acknowledgement Options. Oct. 1996.

[5] K. Fall and S. Floyd. Simulation-based Comparisons of Tahoe, Reno and SACK TCP. Computer Communication Review, V. 26 N. 3, Jul.1996, pp. 5-21.

[6] S. Floyd and T. Henderson. RFC 2582: The NewReno Modification to TCP's Fast Recovery Algorithm. Apr. 1999.

[7] V. Paxson and M. Allman. RFC 2988: Computing TCP's Retransmission Timer. Nov. 2000.

[8] Seddik-Ghaleb,Ghamri-Doudane, Senouci, " Effect of Ad Hoc Routing Protocols on TCP Performance within MANETs", IEEE Sensor and Ad Hoc Communications and Networks,pp.866-873, SECON '06

[9] Qianwen Lin Kwang-Mien Chan Kean-Soon Tan, "Performance analysis of ad-hoc networks partitioning on TCP",IEEE Vehicular Technology Conference, pp.1550-1552, 2005.

[10] R.Caceres and L.Iftode. Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments IOEEE JSAC, 13(5),June 1994.

[11] D.Johnson, D.Mltz, Y.-c.Hu, and J.Jetcheva, "The Dynamic source routing protocol for mobile ad hoc networks," in IETFInternet Draft(work in progress), Nov. 2000

[12] C.Perkins, E.Royer and S Das, "Ad-hoc on demand distance vector(aodv) routing," in IETF Internet Draft(work in progress), Nov.2000.

[13] Berger,Lima,Manoussakis,Pulgarin,Sanchez, " A Performance comparison of TCP protocols over Mobile Ad hoc networks', IEEE Electronics, Robotics and Automotive Mechanics Conference, Vol.2 pp.88-94, 2006.

[14] Sung Jae Jung Mun Gi Kim Byung Ho Rhee, "Performance Enhancement Technique for TCP over Mobile Ad Hoc Networks", IEEE Hybrid Information Technology, pp.210-215, 2006.

[15] Caihong Kai Yuzhong Chen Nenghai Yu, "An Improvement Scheme Applied to TCP Protocol in Mobile Ad Hoc Networks", IEEE Mobile Technology, Applications and Systems, 2nd International Conference, pp.1-6, 2005.

[16] Ye Bin Hu Gu Yu , "The analyze and improve TCP performance using a DSR route protocol based on signal strength", IEEE Wireless Communications, Networking and Mobile Computing, pp. 846 – 849, 2005.

[17] Dongkyun Kim , Hanseok Bae, Jeomki Song, "Analysis of the interaction between TCP variants and routing protocols in MANETs", IEEE Parallel Processing, ICPP 2005 Workshops, pp 380-386, 2005.

[18] Prabakaran, M. Mahasenan, A. , "Analysis and enhancement of TCP performance over an IEEE 802.11 multi-hop wireless network: single session case", IEEE International Conference on Personal Wireless Communications, pp-29-33, 2005

[19] Caihong Kai Yuzhong Chen Nenghai Yu, "An Improvement Scheme Applied to TCP Protocol in Mobile Ad Hoc Networks", IEEE International Conference on Mobile Technology, Applications and Systems, pp.1-6, 2005