

Performance Improvement in MANETs: A Cross Layer Approach via TCP

Jennifer.S.Raj¹, S.Smys², G.Josemin Bala³

^{1,2}Department of Electronics and Communication Engineering, Karunya University, Coimbatore, India. jennionnet@gmail.com

³Department of Electronics and Media Technology, Karunya University, Coimbatore, India.

{Smys, josemin@karunya.edu}

Abstract - Mobile Ad Hoc network is a collection of mobile nodes that are arbitrarily located in such a manner that the interconnections between nodes are capable of dynamically changing on a continual basis. Reliable transport protocols such as TCP are tuned to perform well in traditional wired networks where packet losses occur mostly because of congestion. In wireless link packet losses (due to noise) are often more significant than congestion losses. Congestion avoidance results in a very low utilization of the link when there is an appreciable rate of losses due to link errors. This issue is very significant for wireless links. In this paper to overcome network performance problems, a cross layer design which helps in sharing the information while the layers separation at the design level is still maintained is proposed. This minimizes data retransmissions and losses by analyzing and reacting appropriately to different events occurring at lower layers. The route failure notification from MAC to network layer and the route change notification from network layer to MAC is brought up to the TCP layer, to evolve an efficient and reliable transport protocol in Ad Hoc networks. The mobile network simulator GLOMOSIM is used to study the performance of the network.

Keywords: cross layer, TCP, link failure, MANET.

I. INTRODUCTION

In cellular wireless networks, there are frequent breaks in communication due to handoffs, during which there is packet loss in the transition phase. This leads to the initiation of congestion control procedures at the source and loss of throughput. In addition, wireless links have lower bandwidth and are less reliable than wired links. This adversely affects the end-to-end performance of transport protocols like TCP. In this section a simple approach used in improving the TCP performance [1] is discussed clearly. Ad Hoc networks consist of a set of mobile hosts communicating among themselves using wireless links, without the use of any other communication support facilities. They are also called mobile radio networks or multi hop wireless networks. Two mobile hosts are said to be within range and neighbors of each other if each one can receive others transmission. Every MH (Mobile Host) behaves in a co-operative fashion by acting as a router, allowing packets destined to other MHs to pass through it. The topology of an Ad Hoc network changes every

time as MH's movement results in the establishment of new wireless links or link disconnections. The rate of topology change is dependent on the extent of mobility and transmission range of the hosts. Routes are heavily dependent on the relative location of the MHs. Hence, routes may be repeatedly invalidated in an unpredictable and arbitrary fashion due to the mobility of hosts. The mobility of a single node may affect several routes that pass through it.

It is desirable to use TCP directly even in Ad Hoc networks in order to provide seamless portability to applications like file transfer, email and web browsers written using standard libraries. Hence, it is of interest to study the behavior of TCP [2] in the context of Ad Hoc networks and evaluate the effect of dynamic topology on TCP performance.

The dynamic nature of topology in Ad Hoc networks poses many interesting problems in the domain of routing protocols. In these networks, it is necessary to have a routing protocol that quickly provides relatively stable, loop free routes and adapts to the mobility of the network.

Conventional protocols like LSR, DVR do not match these requirements because they do not converge quickly enough or scale well as mobility increases. A number of alternative protocols have been proposed including a dynamic sequenced distance vector (DSDV), TORA, routing using a virtual backbone, zone routing, cluster-based routing and tree-multicast-based routing strategies. In understanding the effect of packet loss on TCP performance we make the following assumptions:

1. An Ad Hoc network consists of n MHs, each equipped with wireless communication capability. Each MH broadcasts its packets on the wireless channels.
2. The wireless links are bidirectional. This implies that all the MHs have the same transmission range. Otherwise, the weaker hosts can receive the transmissions of stronger hosts but not vice versa.
3. A reliable data link layer protocol is implemented over the unreliable wireless links hence if a packet cannot be delivered at the link layer, the network layer will be duly informed.
4. A suitable routing protocol is implemented to establish and maintain routes between a source and a destination. The routing protocol may maintain redundant routes between different sources and destinations.
5. When a packet cannot be sent on a link despite the presence of a reliable link layer protocol, we treat the situation as the failure of the link due to mobility. The routing protocol is then responsible for adapting and maintaining all routes.

When a route failure occurs, a finite time elapses until the route is restored and communication can be resumed.

6. All packets carry the source and destination ids so that the network layer can identify the source and destination address of each packet.

TCP is a reliable, stream-oriented transport layer protocol which was designed for use over fixed low-error networks like the Internet. Route failures and disruptions are very infrequent since the network is fixed. Therefore, packet loss, which is detected by TCP as a timeout, can be reliably interpreted as a symptom of congestion in the network. In response, TCP invokes congestion control mechanisms. Thus, TCP does not distinguish between congestion and packet loss due to transmission errors or route failures. This inability of TCP to distinguish between two distinct problems exhibiting the same symptom results in performance degradation in Ad Hoc networks.

In an Ad Hoc network, packet losses are frequent in the error-prone wireless medium, but the effect of these losses can be reduced using reliable link layer protocols. The more serious problem is that of route failures, which can occur very frequently and unpredictably during the lifetime of a transport session, depending on the relative motion of a MHs in the network. In general, whenever the mobility of an MH invalidates a route, the reestablishment of the route by the underlying routing protocol takes a finite amount of time. During this period of time, no packets can reach the destination through the existing route. This in turn leads to timeouts at the source which are interpreted by the transport protocol as congestion.

Retransmits unacknowledged packets upon timing out and invokes congestion control mechanisms that include exponential back off of the retransmission timers and immediate shrinking of the window size, thus resulting in the reduction of the transmission rate, and enters a slow start recovery phase to ensure that the congestion is reduced before resuming transmission at the normal rate.

This is undesirable for the following reasons:

1. Packet retransmission wastes precious MH battery power and scarce bandwidth.

This period immediately follows restoration of route; the throughput will be unnecessarily low as a result of the slow start recovery mechanism, even though there may be no congestion in the network. Ad Hoc networks are completely wireless networks [3] of mobile hosts, in which the topology rapidly changes due to the movement of mobile hosts. This frequent topology change may lead to sudden packet losses and delays. Transport protocols like TCP [4], which have been designed for reliable fixed networks, misinterpret this packet loss as congestion and invoke congestion control, leading to unnecessary retransmissions and loss of throughput. When a route is disrupted, the source sends a Route Failure Notification packet, allowing it to invalidate its timers and stop sending packets. When the route is reestablished the source is informed through a Route Reestablishment Notification packet, upon which it resumes packet transmissions. In the event of route failures, the route reestablishment takes place. When a packet loss occurs and the TCP goes for retransmission, the congestion window of TCP is set to finite state. Meanwhile the route reestablishment

occurs when the packet loss is due to route failures. So we prevent the TCP from invoking congestion control algorithms. This provides significant improvements in performance [5], [6].The following diagram.1 depicts the flow of data from the physical layer to the transport layer in the normal sequence through the various event handlers without any interruption,

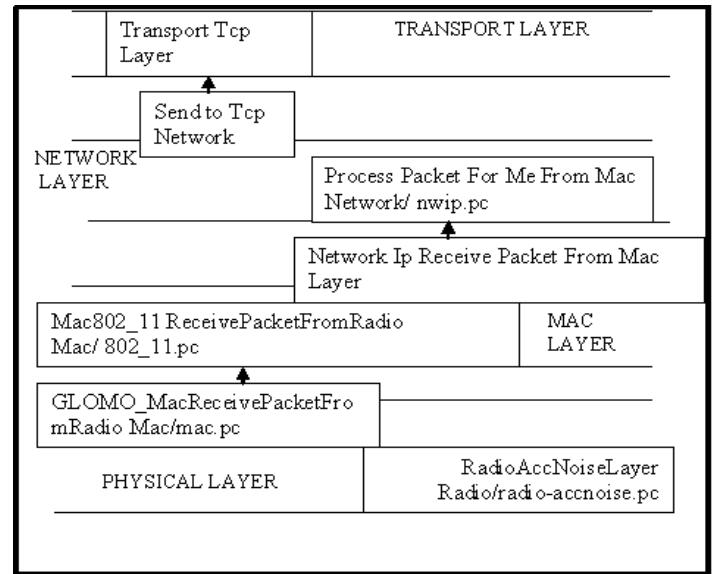


Figure.1 Normal Data Flow

We use the algorithm TCP Veno and identify the packet loss criterion. Once the packet loss is identified due to route failure we invoke the route error packet to mark the route invalid and also send a information to the TCP to bring it to the F-TCP and invoke the route request packet and validate new route for transmission. All this takes place in a span of 400 ms which is the freezing time of TCP. The diagram.2 depicts the flow of data from the physical layer to the transport layer during unfavorable circumstances like loss of data or loss of acknowledgement.

The remainder of the paper is organized as follows. In section 2, the proposed methodology is given in detail. Section 3 and 4 shows the testing parameters and simulation results. Finally section.5 concludes the paper.

II. PROPOSED METHODOLOGY- CROSS LAYER APPROACH

It is clear that treating route failure as congestion is not advisable because congestion and route failure are disparate phenomena which to be handled independently and separately. Therefore, a scheme is proposed by which a source is informed of the route failure so that it does not unnecessarily invoke congestion control and refrain from sending any further packets until the route is restored.

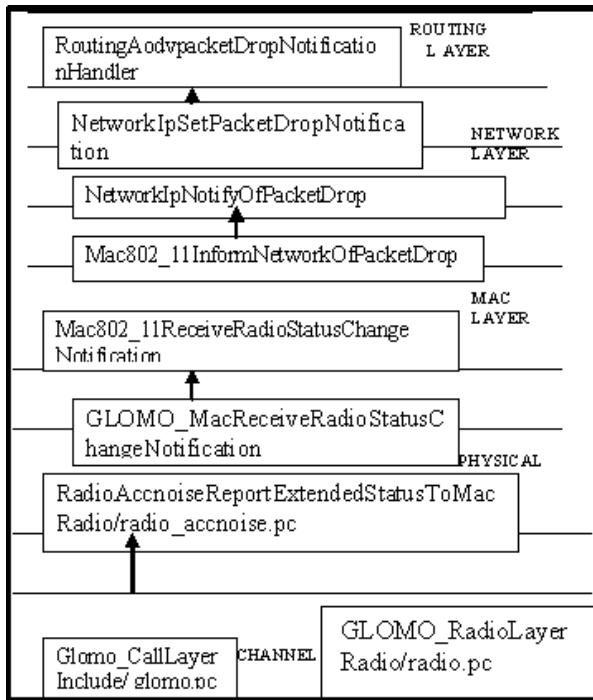


Figure.2 Data flow when a packet is dropped from channel to network layer

Consider a bulk data transfer session, where a source MH is sending packets to a destination MH. As soon as the network layer at an intermediate MH detects the disruption of a route due to the mobility of the next MH along that route, it explicitly sends a route failure notification (RFN) packet to the source and records its event. Each intermediate node that receives the RFN packet invalidates the particular route and prevents incoming packets intended for the destination from passing through that route. If the intermediate node knows of the alternate route to the destination, this alternate route can now be used to support further communication, and the RFN is discarded. Otherwise, the intermediate node simply propagates the RFN toward the source. On receiving the RFN the source goes into the snooze state and performs the following

- It completely stops sending the packets
- It then marks all of the existing timers as invalid
- Freezes the send window of packets
- Freezes the values of other state variables such as retransmit timer value and window size
- Start a route failure timer which maintains to a worst case route reestablishment time.

We have taken the value as 500 ms as it is the time out value in TCP session maintenance timer.

The source remains in this snooze state until it is notified of the route through a route reestablishment notification packet. In Freeze – TCP, when a receiver senses an impending disconnection, it sends a few ACKs with advertised window size equal to 0, thereby freezing the source without reducing the source’s window. As soon as the connection is reestablished, fast transit is enabled by triple ACK. Let one of the intermediate nodes that has previously forwarded an RFN to the source learns about a new route to the destination. This intermediate node then sends an RRN (Route Request

Notification) packet to the source. All other node that receives the RRN simply forwards it to the source.

As soon as the source receives the RRN, it changes to an active state from the snooze state. It then flushes out all unacknowledged packets in its current window. Since most packets in transit during the failure period would have been affected, packets can be flushed out without waiting for acknowledgements from the receiver. The number of retransmitted packets directly depends on the current window size. These steps in effect reduce the effect of TCP’s congestion [7] control mechanism when transmission restarts. Communication now resumes at the same rate ensuring that there is no unnecessary loss of throughput in this period. Various simulation experiments have been conducted to compare the performance of TCP [8] over the proposed one. The results prove that this approach is indeed beneficial in preventing performance degradation. It’s found that as we proceed on this approach the delay parameter is increased. Because we are able to reroute and establish the new path only on delays sacrifice. To overcome this venture we are to adopt various algorithms. Initially when the loss of packets is identified by the data link layer the packet loss detection algorithm TCP Veno is used. As soon as TCP Veno detects the packet loss as route failure we request the AODV to mark the route as error route and then TCP’s congestion window and data link layer’s sliding window are freeze by using the concept of finite state machine. Due to this approach the delay factor is considerably reduced.

III. TESTING PARAMETERS

For our experiments we have used a fixed packet size of 5000 bytes. The communication is performed between 9 pairs of nodes and they are taken as data1 to data9. The input parameters to the simulation are given below.

Inputs file description.

TERRAIN-RANGE-X	100
TERRAIN-RANGE-Y	100
POWER-RANGE	50
SEED	1
SIMULATION-TIME	100M
NUMBER-OF-NODES	1
NODE-PLACEMENT	RANDOM
BANDWIDTH	2000000
MAC-PROTOCOL	802.11
ROUTING-PROTOCOL	AODV
NETWORK-PROTOCOL	IP
TRANSPORT-PROTOCOL-TCP	YES
TRANSPORT-PROTOCOL-UDP	YES
TCP-STATISTICS	YES
UDP-STATISTICS	YES
ROUTING-STATISTICS	YES
NETWORK-LAYER-STATISTICS	YES
MAC-LAYER-STATISTICS	YES
RADIO-LAYER-STATISTICS	YES
CHANNEL-LAYER-STATISTICS	YES
MOBILITY	RANDOM-WAYPOINT
MOBILITY-WP-PAUSE	30S
MOBILITY-WP-MIN-SPEED	0

```

MOBILITY-WP-MAX-SPEED 10
MOBILITY-POSITION-GRANULARITY 0.5
APP-CONFIG-FILE app.conf
FTP <source> <dest> <items to send> <start time>
TELNET <source> <dest> <session duration> <start time>
FTP: <items to send>
TELNET: <session duration>
    
```

IV. SIMULATION RESULTS

The source sends a continuous stream of data to the destination through the simulated Ad Hoc network. The effect of topology change of the network from the point of view of a transport entity and abstracts the other aspects of the network layer routing. The route reestablishment delay reflects the ability of the network to respond to topology changes. In Figure 3, the normal performance of the network is displayed. The graph is plotted between the mobility of the nodes and its throughput performance in various channel conditions. The data's 1 to 9 corresponds to various noise figures 10 to 90. It's observed that when the size of the packets are increased the throughput decreases.

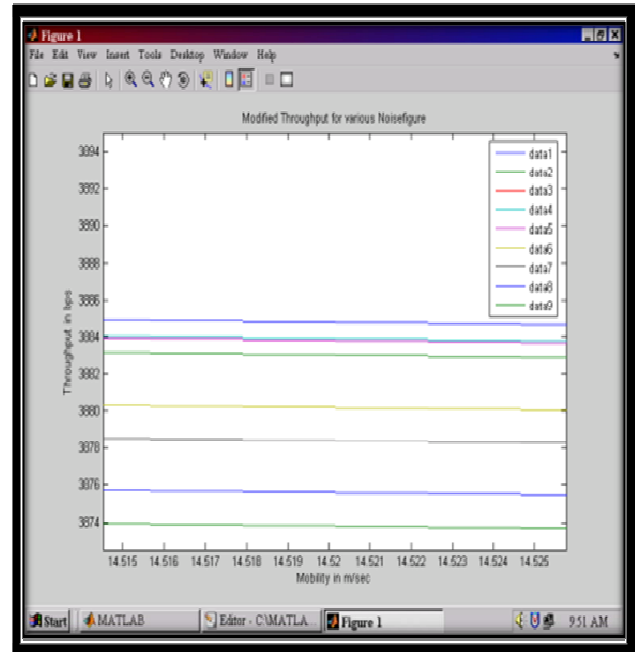


Figure.4 Mobility vs. Throughput for various packet sizes.

In Figure.5 shows the end to end delay remains constant for all transmissions irrespective of the channel conditions. The channel condition is made worse by varying the noise figure. The graph is plotted between the mobility of the nodes and its delay performance in various channel conditions. The data's 1 to 9 corresponds to various noise figures 10 to 90.

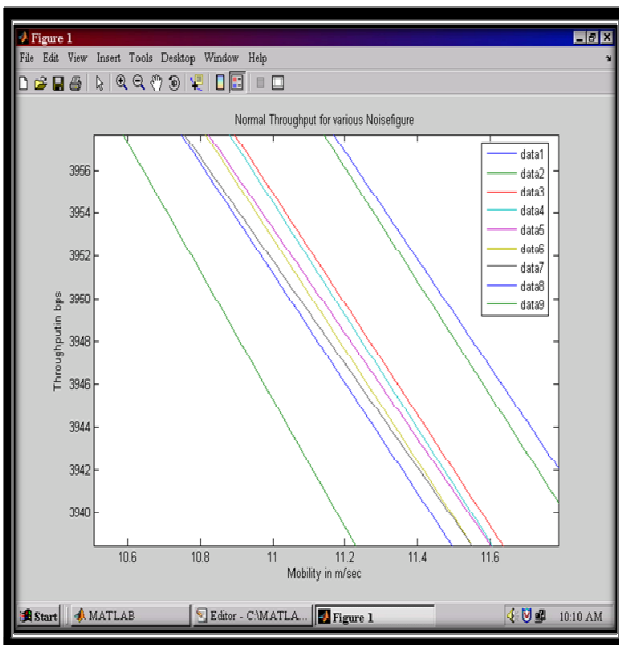


Figure.3 Mobility vs. Throughput for various packet sizes.

Also we infer that the throughput falls when the node mobility increases. Figure.4 we find that throughput of the system is at most constant with varying packet sizes and node mobility. The graph is plotted between the mobility of the nodes and its throughput performance in various channel conditions. The data's 1 to 9 corresponds to various noise figures 10 to 90.

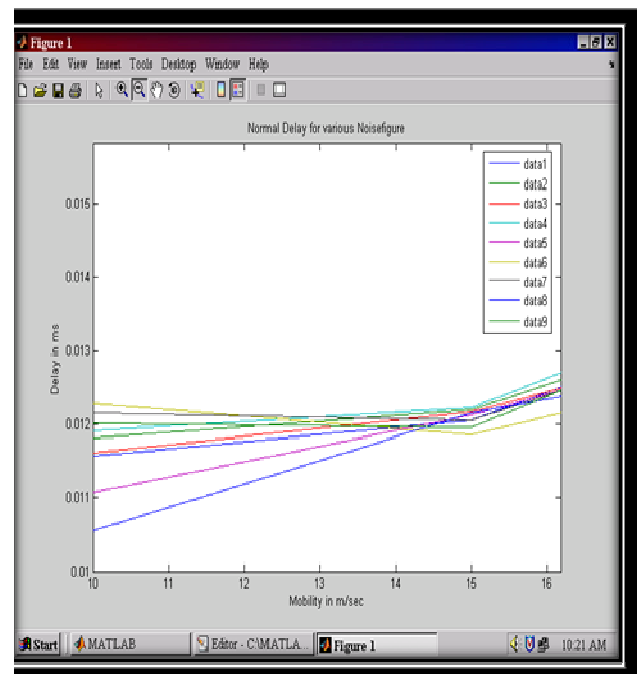


Figure.5 Mobility vs. Delay for various packet sizes.

Delay for various packet sizes during mobility was displayed in Fig 6. From the results its clearly understood that, end to end delay is increased as the channel conditions are made worse. The graph is plotted between the mobility of the nodes and its delay performance in various channel conditions. The data 1 to

9 corresponds to various noise figures 10 to 90. To make the analysis more accurate we found the throughput and delay performance of the various noise figures of a fixed packet size on applying the proposed idea and the existing scheme.

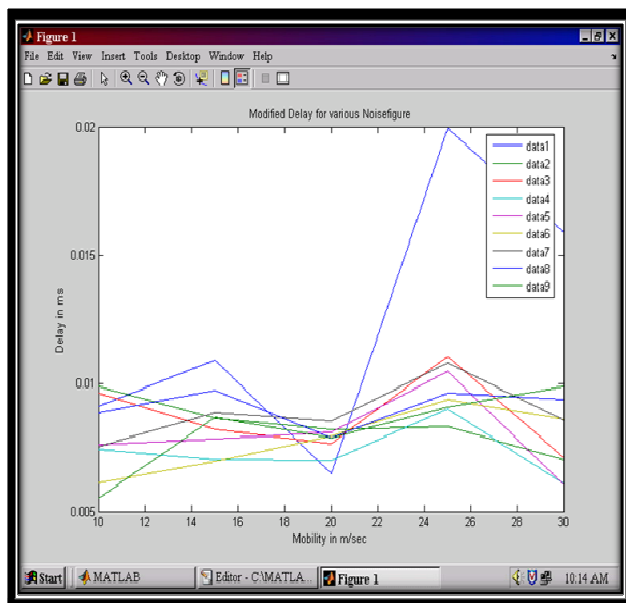


Figure.6 Mobility vs. Delay for various packet sizes.

V. CONCLUSION

In TCP, if the source is not aware of the route failure, the source continues to transmit packets even when the network is down. This leads to packet loss and performance degradation. Since packet loss is interpreted as congestion, TCP invokes congestion recovery algorithms when the route is reestablished, leading to throttling of transmission. The failure point notifies the source of route failures and route reestablishments thus distinguishing route failures and route reestablishments, thus distinguishing route failures from congestion. As the route reestablishment delay grows, the performance increases significantly. This is attributed to the fact that we are able to prevent unnecessary timer back. When a packet loss occurs there is an end to end retransmission in TCP. In this paper we have stopped the retransmission and freeze the window of TCP at receiver as well as sender.

It happens in lower layers, in the same time rerouting and the fresh route is determined. After the determination of fresh route the packet is forwarded in that path hence the throughput of TCP is improved. We expected that as the delay increase the throughput will be improved much. But from the simulation results we observed that the delay is not increased due to the 2msl timer which runs after all actions. If this too is stopped the performance can be further improved a lot.

REFERENCES

[1]. Kathik Chandran, "A feedback-based scheme for improving TCP performance in AdHoc Wireless Networks", *IEEE Personal Communications*, 34-39, Feb 2001.

[2]. A. Chockalingam, Michele Zorzi and Velio Tralli, "Wireless TCP Performance with Link Layer FEC/ARQ", in *Proc. ICC'99*, 1999, pp. 1212-1216.

[3]. Rami G. Mukhtar, Stephen V. Hanly, and Lachlan L. H. Andrew, University of Melbourne, "Efficient Internet Traffic delivery over Wireless Networks", *IEEE Communications Magazine*, pp. 46-53, Dec 2003.

[4]. Seoung-Bum Lee, Gahng-seop Ahn and Andrew T. Campbell, "Improving UDP and TCP performance in Mobile Ad hoc Networks with INSIGNIA", *IEEE Communications Magazine*, pp. 156-165, June 2001.

[5]. Cheng Peng Fu, Associate Member, IEEE, and Soung C. Liew, Senior Member, IEEE, "TCP Veno: TCP Enhancement for Transmission Over Wireless Access Networks", *IEEE Journal on Selected Areas of Communication*, Vol. 21, No. 2, pp. 216-228, Feb 2003.

[6]. Hari Balakrishna, Student Member, IEEE, Venkata N. Padmanabhan, Student Member, IEEE, Srinivasan Seshan, and Randy H. Kantz, Fellow, IEEE, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links", *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, pp. 756-769, Dec 1997.

[7]. N. K. G. Samaraweera, "Non-congestion packet loss detection for TCP error recovery using Wireless links", *IEEE Proc. Commun.*, Vol. 146, No. 4, pp. 222-229, August 1999.

[8]. [Katz95] Balakrishna H, Katz R. et al "Improving TCP/IP performance over wireless networks" *Proc. 1st ACM Int'l Conf on Mobile Computing and Networking (Mobicom)*, Nov 1995.