TCP PERFORMANCE FOR FUTURE IP-BASED WIRELESS NETWORKS

Deddy Chandra and Richard J. Harris
School of Electrical and Computer System Engineering
Royal Melbourne Institute of Technology
Melbourne, Australia 3000
[deddy.chandra@ieee.org] [richard@catt.rmit.edu.au]

Nirmala Shenoy
Department of Information Technology
Rochester Institute of Technology
Rochester, NY 14623
[ns@it.rit.edu]

Abstract - In recent years, many extensive studies have been performed that have attempted to improve the performance of TCP for wireless networks. The common assumptions made by TCP for wired networks definitely do not hold for wireless networks. TCP with its semantics such as end-to-end flow control, a congestion control mechanism and error recovery provide reliability in wired networks. Wireless communication has significantly different characteristics compared to wired networks such as higher bit error rates, higher latency, limited bandwidth, multi-path fading of the signals and handoff. In this paper, we propose an enhancement to TCP that will be referred to as E-TCP, which improves upon conventional TCP congestion and flow control in a wireless environment. Our simulation results show significant improvements to TCP performance can be achieved.

Keywords: TCP, wireless network, flow control, congestion control.

I. Introduction

The Internet and wireless networks are two of the most important technical developments that have had a direct affect on the lives of people over the past few years. The next step in these developments will be to combine these two technologies to provide wider-area wireless Internet access [10].

The Transport Control Protocol/Internet Protocol (TCP/IP) [8][9] is the protocol suite that integrates a wide range of different physical networks into a global Internet. TCP was developed to perform well in traditional wired networks where TCP serves as a transport protocol that guarantees a reliable service for sending data over a network. “Reliable service” in this context means service that will ensure that the destination successfully receives data that has been sent by the source.

Wireless media have different properties from those upon which the design of TCP was based. In particular, TCP assumes that network links rarely corrupt data [6], because the loss of data is taken as a signal for congestion in the network. Thus, upon the loss of a data segment, TCP reduces its sending rate to alleviate congestion in the network. This approach is effective if data losses are caused by congestion in the network. However, wireless media have higher bit error rates and this leads to significant corruption losses. This will cause TCP to unnecessarily reduce its transmission rate, yielding a significant degradation in performance.

The poor performance of TCP in error-prone wireless networks is mainly due to a lack of knowledge by TCP about the real cause of packet loss. To improve its performance, TCP needs an explicit acknowledgment about the cause of packet loss. Thus, TCP is able to take appropriate action in the case when packet losses have been detected. In wireless networks, packet loss is usually due to corruption; therefore it is unnecessary for TCP to reduce its transmission rate. Hence, it will be able to maintain its throughput.

Generally, there are two different approaches to improve TCP performance in a wireless network. The first approach attempts to hide any non-congestion related losses from the TCP sender and requires no changes to the existing implementation of the TCP sender. The reasoning behind this idea is that since the problem is local, the problem should be solved locally. As a result, most of the losses seen by the sender are due to congestion [3]. The second approach attempts to make the TCP sender aware of the existence of a wireless environment, and aware that some packet losses are not due to congestion. The sender can then avoid invoking the congestion control mechanism when non-congestion related losses occur.

Many proposed schemes and mechanisms try to improve TCP performance over wireless networks. But none of them lets TCP clearly know the cause of packet loss, which results in a significant degradation of TCP performance. The proposed schemes can be classified into three categories; they are end-to-end schemes [4][5], link layer schemes [3] and split connection schemes [1][2].

Our objective is to improve TCP performance over wireless networks. Our work here follows the second general approach discussed above, which is to make the TCP sender aware of the existence of a wireless environment in the connection. In this paper, we propose an acknowledgment scheme together with an agent on the intermediate node that is located between the wired and the wireless network. This scheme will help TCP to distinguish between congestion and corruption loss. More functionality is required at the base station as well as at the TCP sender to deal with determination of congestion

and corruption losses. We show how to implement our proposed scheme and demonstrate its effectiveness through simulation.

The remainder of this paper is organized as follows. Section II describes the assumptions concerning the internetwork structure. Our proposed solution is completely presented in section III. A simulation model and results will be discussed in section IV. Conclusions and future work are presented in section V.

II. Internetwork Structure

We assume that our internetwork model consists of multiple networks that are joined together by computers attached to one or more networks that are known as routers. Data is delivered from a computer in one network to another computer in another network via a router. The data is forwarded from one intermediate node to another until it reaches its final destination.

As shown in Figure 1, the internetwork may consist of a group of wired networks that form the core of a wired network and a wireless networks. We may assume that data are relatively corruption free in the wired network. Thus, it is reasonable to assume that in the wired network data might be lost due to buffer overflows, i.e. congestion. Wireless networks are not allowed to serve as intermediate hops in the routing path between two networks [7]. Two or more computers in the internetwork can establish a TCP session between them, regardless of whether they are mobile or fixed host.

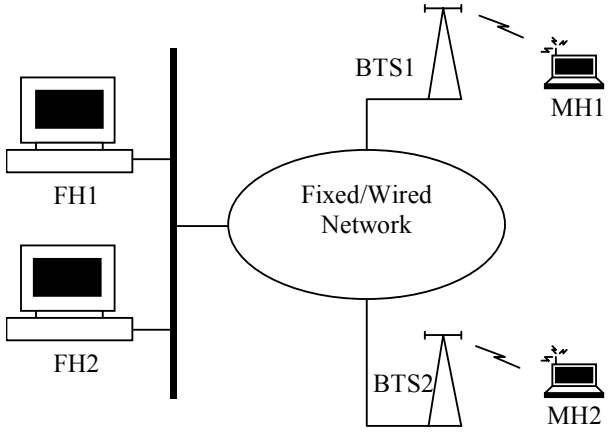

Figure 1: Internetworking Model

We assume that routers provide only data delivery; therefore they do not guarantee its delivery. Routers forward each data packet to the next computer or router and this machine may drop the data due to buffer overflow (congestion). Routers in wired networks are not aware of mobile hosts and must not be affected by any changes that support wireless communication. We assume that enhancing the functionality of wireless routers to a low degree is acceptable. However, requiring a wireless router to reliably deliver data is too much to ask, due to the processing overhead and violation of TCP semantics. Routers that are attached to wireless networks are aware that a wireless network is prone to corruption and are located at the edge of the internetwork (between the wired and wireless networks). Thus, we found this router is the best point to use to help the TCP sender to determine the cause of packet loss.

III. Enhanced-TCP (E-TCP)

We propose a new enhanced type of TCP called E-TCP (Enhanced TCP) that can differentiate packet losses either due to corruption or congestion and take appropriate action to overcome the losses. We also introduce an implementation of a transit agent (TA) at the gateway node[1]. A transit agent will explicitly acknowledge E-TCP for the travel history of every packet that has transited at the base station. Thus this acknowledgement will become information for E-TCP to take appropriate recovery action over packet losses. This scheme needs modifications to the structure of the acknowledgment packets in such a way that it will contain up to two bits of information (assuming that communication between two mobile hosts involves two supporting base stations), the software part of the transport protocol of the base station and the sender side host. In most proposed schemes, they are only concerned with communication between the fixed host and the mobile host, but our proposed scheme not only can support communication between a fixed host and a mobile host, but it also supports communication between mobile hosts.

Table 1: Cross-communication between hosts

xmt \ rcv	Fixed Host	Mobile Host
Fixed Host	TCP	E-TCP
Mobile Host	E-TCP	E-TCP

A. New Acknowledgment packet

In our proposed scheme, we are going to use a new form of acknowledgment packet, which will be referred to as $ACK_{C\text{-}CLN}$ (congestion-corruption loss notification). This acknowledgment packet will need to use up to two additional bits in the TCP header to store information from the transit agent. We need to allocate two from the six reserved bits in the TCP header for this purpose. When communication occurs between a fixed host and a mobile host, data will only transit through one base station. Thus, one bit will be used. If communication occurs between mobile hosts that might involve two base stations (Figure 1), two bits will be used instead, to acknowledge the sender for a lost packet. For simplicity of understanding, we name each bit in the $ACK_{C\text{-}CLN}$ packet, the first bit will be called "Ob" (Originating BTS) and the second bit called "Db" (Destination BTS). The

[1] A node that sits at the gateway between the wired and wireless networks. For simplicity of understanding, we assumed here that a base station is the gateway node.

base station that serves as a data sender for transmission will use the Ob bit and base station that serves the data receiver will put its information in the Db bit.

B. Transit Agent at the Base Station

By using the concept of an agent at the base station, the agent that we propose here needs to be permanently allocated on the base station and it is only required to perform a simple task. The agent needs to record the sequence number of every packet that transits through the base station before they get forwarded to the receiver. The flowchart for data processing by the transit agent is shown in Figure 2. The agent will also acknowledge the sender of the transit history of packet that is requested by the receiver through the use of $ACK_{C\text{-}CLN}$. Upon receiving an $ACK_{C\text{-}CLN}$ packet from the receiver, the agent will check its record and set a value 0 or 1 in the $ACK_{C\text{-}CLN}$ based on the record. A "0" in the $ACK_{C\text{-}CLN}$ that indicates the expected sequence number has never transited the base station before and a "1" indicates that the expected sequence number has transited the base station.

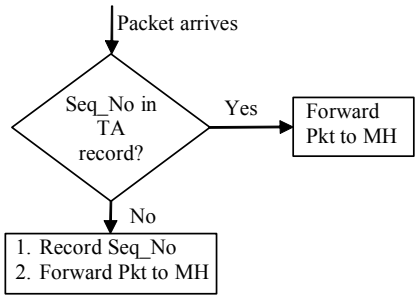

Figure 2: Flowchart for data processing in BTS

C. Transport Protocol at Sender Side

At the sender side, we propose an enhanced version of TCP that has more functionalities than conventional TCP and we call it E-TCP (Enhanced TCP). E-TCP has additional functionality to read information that has been inserted into the $ACK_{C\text{-}CLN}$ packet by the transit agent and takes appropriate packet lost recovery action based on the given information. The implementation of current TCP (TCP Reno) will still be used in the case where both ends are on the fixed network (in this case the sender does not need to use E-TCP since wireless network is not involved in the data transmission and the only reason for packet lost would be network congestion). Therefore, the protocol stack shown below will be implemented on the sender side to improve TCP performance.

There will be two available transport protocols to maintain reliability of data transmission between the two ends depending on the type of network node at each end. Once the sender knows about the type of the receiver, the sender will use TCP or E-TCP respectively. The usage of both protocols is depicted in Table 1.

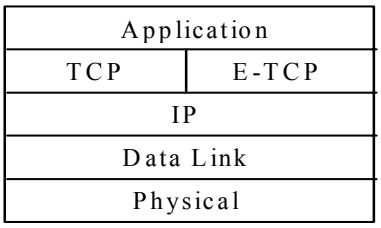

Figure 3: Proposed Protocol Stack

D. Scenarios

In our proposed protocol stack, (Figure 3) we have both TCP and E-TCP located on the transport layer. The type of sender and receiver will determine which one should be used. After the handshake process, the sender will be able to choose which transport protocol is appropriate to handle the data transmission. There are four possible communication scenarios that could occur between hosts as shown Table 1. We shall discuss all four possibilities of communication between them below:

D.1. Fixed host to Fixed host

Data transmission in this direction has been discussed many times and many improvements have been suggested; therefore we leave the current implementation of TCP to handle the reliability of data transmission between fixed hosts. It is unnecessary to use E-TCP in this case, since wireless network does not involve data transmission between fixed hosts. Therefore, any packet losses could be assumed to be congestion loss.

D.2. Fixed host to Mobile host

In this direction of data transmission, a sender (fixed host) will be facing a great challenge to determine the cause of packet loss because the data will travel over both fixed and wireless networks. To maintain its performance, TCP needs to know exactly where packets get lost. We discover that a base station that is located right between the two different networks is a perfect location to help the TCP sender determine the cause of packet loss. Hence, we put an agent at the base station to perform a "transit checking" task.

Every time a packet transits the base station, the transit agent will record the sequence number of the packet. This record shows that the packet has successfully travelled over the fixed network, and then the packet is forwarded to the receiver.

Upon receiving an acknowledgment ($ACK_{C\text{-}CLN}$) packet that contains a request for the receiver's expected packet, the agent will check its record for the sequence number of that expected packet. If the agent finds the sequence number in the record, it will set a value of 1 into the "Ob" bit. Otherwise, the agent will set value of 0 into the "Ob" bit.

Upon receiving an $ACK_{C\text{-}CLN}$ packet, the sender knows which packet the receiver expects. It will also check the Ob bit, if it finds that the value equals 1, and then the sender knows that the expected packet has transited the base station but it was not received by the receiver. This means that the packet was lost in the wireless network

(due to corruption). On the other hand, if the sender found the value was 0, this means the expected packet has never transited this base station. Therefore, it responds by indicating that the packet was actually lost in fixed network (due to network congestion).

D.3. Mobile host to Fixed host

In data transmission from the mobile host to the fixed host, the scenario is quite similar to the data transmission scenario for data transmission from the fixed host to the mobile host. The difference is only on the interpretation of the value "Ob" by the sender. When the value of Ob is equal to 1, it means that the expected packet has transited the base station, thus the expected packet was lost in the fixed network. On the contrary, if the Ob value is equal to 0 then the expected packet was lost in the wireless environment.

Figure 4 shows the flowchart of $ACK_{C\text{-}CLN}$ processing in the base station.

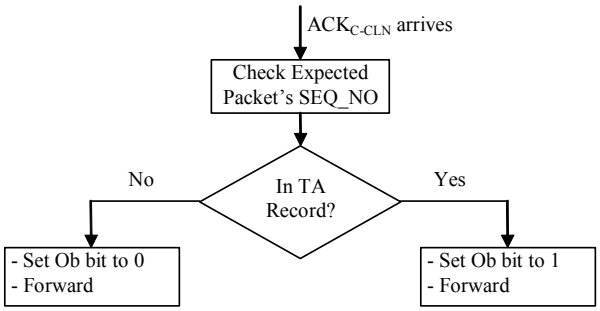

Figure 4: Flowchart of $ACK_{C\text{-}CLN}$ Processing in BTS

D.4. Mobile host to Mobile host

From our system model, we also consider data transmission between mobile hosts. Since a packet must travel twice over the air which might be handled by two base stations and once over a fixed network, it is possible for a packet to get lost in either the wireless network or the fixed network (Figure 1). In the case when the packet loss occurs in the wireless area between the second base station (base station that serves the receiver) and the second mobile host (assume this mobile host is the receiver), the sender has no idea of the cause of packet loss and it will again assume that packet loss is due to congestion. Therefore, the $ACK_{C\text{-}CLN}$ will need to have another bit named "Db" which will contain confirmation of the transiting packet set by the second base station. In this case, as previously stated, the second base station will put its confirmation about the transit status of the "transiting packet" in the Db bit. When this $ACK_{C\text{-}CLN}$ has reached the first base station, this base station will assign a value on its Ob bit. With these two bits of information, the sender will be able to determine where the losses have occurred. We can identify a number of different cases and they are identified below:

Case 1: Ob = 0 and Db = 0

This shows that the expected packet was lost in the wireless environment between the sender and the first base station. In other words the packet was lost in the wireless environment while it was on its way to the first base station. Therefore, the lost packet needs to be retransmitted. But, the sender does not need to reduce its congestion window size.

Case 2: Ob = 1 and Db = 0

In this case, the packet was lost in the area between the first and the second base station. This means that packet was lost in the fixed network. Hence, the sender needs to retransmit the lost packet and deploy its congestion avoidance mechanism.

Case 3: Ob = 1 and Db = 1

In this case, the packet was lost in the wireless environment between the second base station and the receiver. In order to recover, the sender must retransmit the lost packet but then it does not need to reduce its congestion window size.

Figure 5 shows the flowchart for $ACK_{C\text{-}CLN}$ processing in the BTS for mobile-to-mobile communication.

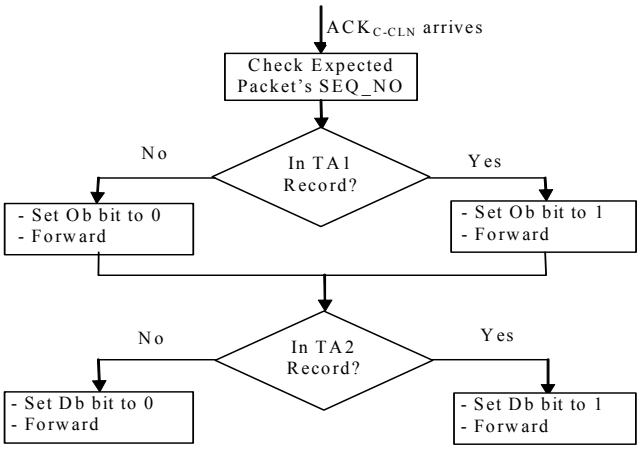

Figure 5: Flowchart of $ACK_{C\text{-}CLN}$ processing for mobile-to-mobile communication

IV. Simulation

This section describes a simulation study based three different scenarios for data transmission as depicted in Figure 1. The sender might be a fixed host or a mobile host and it is a similar case for the receiver. The base station includes a finite-buffer drop tail gateway and the network has both wired and wireless links. It is assumed that the sender always has data to send. It is also assumed that the receiver can always send out acknowledgments immediately for each data received without delay, other than the processing delay.

For simplicity, the simulations shown in this paper use a receiver that sends an ACK for every data packet received. The simulations also consist of one-way traffic. As a result, ACK packets are never "compressed" or discarded on the path from the receiver back to the sender. The simulation model was implemented using the OPNET^{TM} simulator.

The current implementation of TCP (TCP Reno) has been considered in this simulation. The simulation was performed under a typical wireless packet loss environment with bit-error rate and we assume that the bit error rate for wired links is very low. Table 2 summarises some parameters used in our simulation model.

Table 2: Simulation Parameters

Wired link capacity	56 Kbps
Wired link propagation delay	30 msec
Wired link BER	10^{-9}
Wireless link capacity	2 Mbps
Maximum window size	94
TCP segment size	170 bit
Minimum slow-start threshold	1
Bit error rate	10^{-3} to 10^{-6}

In order to measure the performance of TCP over a lossy link, we simulated all models using a range of different values of the bit-error rate (Table 2). All results from our simulations are obtained under 95% confidence levels. We have performed our simulation under three different scenarios. The three scenarios cover the following three cases: We first assume that data was sent from the fixed host to the mobile host. In the second case data was transmitted from the mobile host to the fixed host and at finally data transmission from mobile to mobile host. In all of these cases we assume that the mobile host stays in one cell during the lifetime of the connection. We then report the simulation results under varying bit-error conditions over the wireless channel, where the ranges of BERs are from 10^{-3} to 10^{-6}.

Table 3: Number of received packet at receiver under BER 10^{-3}

direction / protocol	F to M	M to F	M to M
E-TCP	3176	3341	3459
Reno-TCP	2624	2722	2735
%	21.04	22.74	26.47

Performance of E-TCP and Reno on data transmission from fixed host to mobile host is depicted in Figure 6. Under different values of BER, it shows that E-TCP performance is slightly better than Reno as the BER is increased. E-TCP allows the receiver to receive more packets since the sender was able to detect the real cause of packet losses. Thus, the sender was also able to maintain its transmission rate since it found out that the packet losses were due to corruption. Table 3 shows that E-TCP had sent on average about 21.04% more packets compared with Reno-TCP. The reason is that E-TCP could maintain its congestion window size. Therefore, the sender could send packets under an unchanged window size after corruption lost had been detected. On the contrary, Reno-TCP still assumes that lost packets over wireless were due to congestion. Thus, it reduces its congestion window size after the lost packet had been retransmitted and fewer packets will be sent afterward.

Figure 7 shows goodput comparisons of E-TCP and TCP Reno on data transmission from mobile host to fixed host. It also shows that E-TCP could provide a better goodput than Reno. TCP Reno has difficulty in the recovery process since Reno has no indication of the type of packet loss and it assumes that corruption losses are congestion related losses. Therefore, invocation of the congestion control mechanism and reduction of the congestion window size could not be avoided. On the contrary, E-TCP with notification from the transit agent could interpret packet losses that are due to corruption and run its recovery action by retransmitting lost packets without reducing its congestion window size. On average, about 22.74% more packets have been received successfully by implementing E-TCP.

Finally in Figure 8 we show the goodput comparisons under different values of BER between E-TCP and Reno-TCP on data transmission from one mobile to another mobile host. In this figure, we can also see that E-TCP could provide a better goodput compared to TCP Reno. Our simulation results show that 26.47% more packets were received successfully at the receiver.

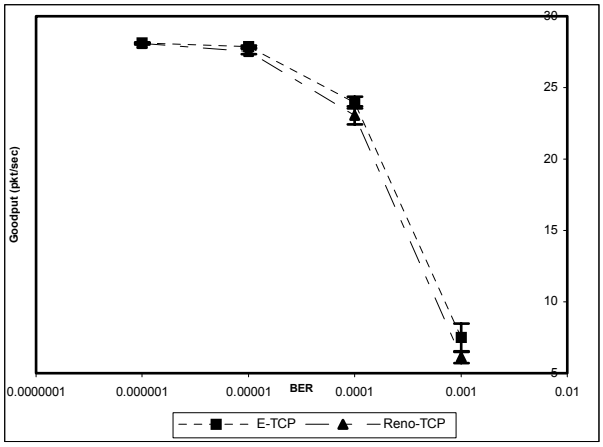

Figure 6: Goodput comparisons at different BER on data transmission from fixed to mobile host

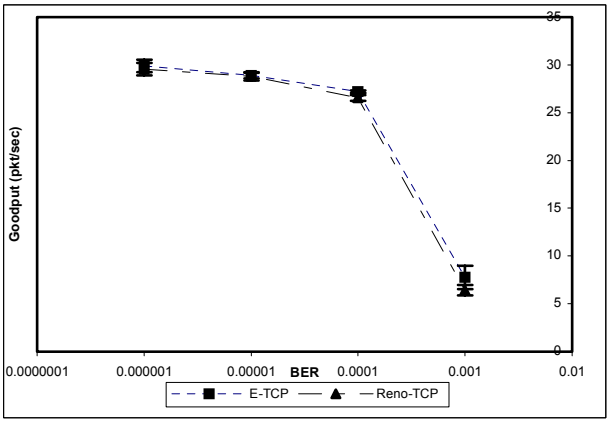

Figure 7: Goodput comparisons at different BER on data transmission from mobile to fixed host

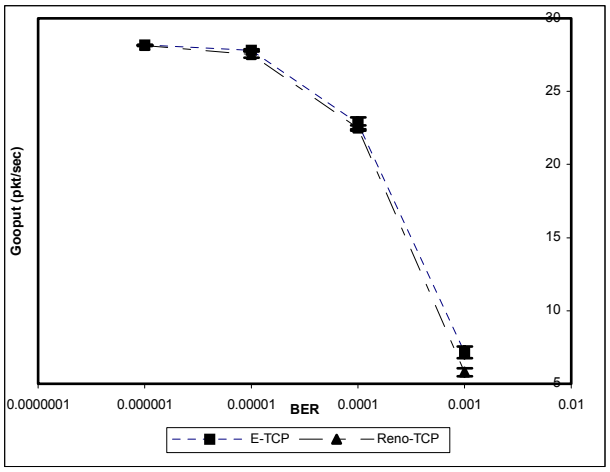

Figure 8: Goodput comparisons at different BER on data transmission from mobile to mobile host

V. Conclusions and Future work

In this paper, we proposed a novel new transport protocol called Enhanced-Transport Control Protocol (E-TCP), $ACK_{C\text{-}CLN,}$ a new form of acknowledgment packet and Transit Agent, an agent that is allocated at a gateway node. Our proposed schemes allow the transport protocol to distinguish between congestion and corruption losses. The transit agent will explicitly acknowledge the sender through ACK packets ($ACK_{C\text{-}CLN}$) of the transit history for every expected packet. The transport protocol at the sender (E-TCP) uses this information to determine the type of packet losses and it adjusts the window size accordingly after the recovery action. This scheme requires only a small amount of functionality at the base station and at the sender, while leaving all other routers in the wired network unchanged. We have shown how to incorporate the scheme into the current implementation of the TCP protocol, and to demonstrate the effectiveness of the approach through simulation.

The key idea here is to explicitly acknowledge the TCP sender in the case where packet losses have been detected, whether data (packets) have arrived at the base station or not. Thus, the transport protocol at the sender will be able to judge the reason for packet losses. This scheme also allows the sender to quickly discover packet loss without having to wait until it has received three duplicate acknowledgments. Furthermore, the sender can quickly retransmit the lost packets. Since our proposed transport protocol could maintain large congestion window size, more packets could be sent to the receiver and throughput could be increased. Simulation results show that our proposed solution provides better results compared with the implementation of conventional TCP (TCP Reno).

In this paper, we have not taken handoff loss into consideration. Our ongoing research will investigate the ability of E-TCP to cope with handoff related losses. We are also investigating any additional improvements that could be adopted to enhance our proposed schemes. Since our proposed scheme includes an acknowledgment scheme, the loss of an acknowledgment packet could be cumbersome. We shall address and investigate this problem in our future work.

REFERENCES

[1] A. Bakre and B.R. Badrinath, "I-TCP:Indirect TCP for mobile hosts," *in Proc. Of 15th Int. Conf. Distributed Computing Syst. (ICDCS)*, May 1995.

[2] K. Brown and S. Singh, "A Network Architecture for Mobile Computing," *in Proc. of IEEE Infocom '96*, pp.1388-1396, 1996.

[3] H. Balakrishnan, V. N Padmanabhan, S. Seshan, and R.H. Katz, "A comparison of mechanisms for improving TCP Performance over wireless links," *IEEE/ACM Trans. On Networking*, vol. 5, no. 6, pp. 756-769, December 1997.

[4] R. Caceres, L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments," *IEEE JSAC*, vol. 13, no. 5, pp.850-857, June 1995.

[5] R.C. Durst, G.J. Miller and E.J. Travis, "TCP extension for space communication," *Wireless Network*, vol. 3, pp. 389-402, 1997.

[6] V. Jacobson, "Congestion Avoidance and Control", *Computer Communication Review*, vol.18, no.4, pp.314-329, August1988.

[7] J.A. Cobb, P. Agrawal," Congestion or corruption? A strategy for efficient wireless TCP sessions", *IEEE Symposium on Computer and Communications (ISCC'95)*, June 1995.

[8] W.R. Stevens, *TCP/IP Illustrated*, Vol. 1. Reading, MA: Addison-Wesley, Nov. 1994.

[9] W. Steven, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithm," *Internet Engineering Task Force*, January 1997, RFC 2001.

[10] W.T. Chen, J.S.Lee, "Some Mechanisms to Improve TCP/IP Performance over wireless and mobile computing environment," *in IEEE Parallel and Distributed Systems Proceedings - Seventh International Conference*, pp: 437–444, July 2000.