Reliable ELN to enhance throughput of TCP over wireless links via TCP header checksum

Pawan Kumar Gupta[1, 2], Joy Kuri[1]
[1]Centre for Electronics Design and Technology (CEDT),
Indian Institute of Science, Bangalore 560012, India
[2]Centre for Development of Telematics (C-DOT), Bangalore 560052, India
Email: pawankumargupta@hotmail.com; kuri@cedt.iisc.ernet.in

Abstract— **In wireless mobile networks, loss of packets is mostly because of corruption of data on the wireless link. TCP responds to these losses as if they are due to congestion, and this results in poor throughput. We suggest enhancements to TCP that, if implemented, will help the receiver in separately identifying corruption losses and congestion losses. Based on this information, the receiver will generate Explicit Loss Notification (ELN) for the sender. We derive an expression for the probability of a receiver generating successful ELN, assuming a generic link layer protocol that is used for data transfer over wireless links. We also suggest modifications to the sender behavior on receiving successful ELN from the receiver. We compare the proposed scheme (called "NewRenoEln") and the standard "NewReno" TCP via simulation, and find considerable improvement in data throughput over wireless links.**

I. Introduction

IN today's world more and more people are moving towards using their mobile devices to access the Internet either for work or entertainment. Transmission Control Protocol (TCP) is the ubiquitous transport protocol used in the Internet world to support applications like telnet, ftp and http [1], [2]. TCP runs above the unreliable and connectionless Internet Protocol (IP) layer and provides connection oriented, end-to-end reliable delivery of application data.

TCP assumes that any loss of packets in the network is due to congestion in the intermediate nodes. This assumption is not valid for wireless networks as they are characterized by large error rates due to fading, noise, interference from other sources and mobile host movement. Thus, in wireless networks more packets are lost due to corruption in the wireless hop as compared to congestion in the wired network. Current TCP implementations respond to any such loss of packet by reducing the congestion window and retransmitting the lost packet. If the loss is due to corruption, then the sender need not invoke the congestion control procedure and just the retransmission of the lost packet will be enough for recovery. On the other hand if the sender does not reduce the congestion window when the loss is indeed due to congestion, then it will result in congestion collapse in the network and will result in more losses and further degradation of throughput. Thus an effective mechanism is required, whereby the sender should be able to identify corruption losses with high probability.

A lot of research has been done in this area, and many authors have suggested different ways of increasing throughput for TCP transmission over wireless networks. The proposals suggested in [3] – [5] mainly focus on hiding corruption losses from the TCP sender by performing retransmissions of any lost data to avoid timeouts. In [6] – [7], the authors have talked about mechanisms to successfully differentiate congestion loss from corruption loss in various scenarios. These schemes rely on intermediate nodes to provide sufficient information for distinguishing corruption loss from congestion loss. Such mechanisms are not useful when the IP traffic is encrypted, or when incorporating TCP-awareness in intermediate nodes is not feasible. In [8] and [9], the authors have suggested the use of successive inter-arrival times of packets to heuristically distinguish corruption loss from congestion loss. These mechanisms work on end-to-end basis but are not very reliable.

In the parallel work done in [10] similar to ours, a new scheme to reliably distinguish corruption loss from congestion loss is suggested. It uses a separate checksum for the TCP header. This is the key for generation of Explicit Loss Notification (ELN) in our proposal as well, but in our analysis we take into account the link level fragmentation of TCP packets into smaller frames by the base station. Our analysis shows clearly that introducing a separate checksum for the TCP header definitely improves the probability of the detection of corruption loss by the receiver. We have simulated this new protocol that we call "NewRenoEln" in Berkeley's network simulator ns. We compare the performances of NewRenoEln and NewReno in the presence of correlated and random errors, and show that NewRenoEln definitely results in better throughput as compared to NewReno over wireless links.

This paper is organized as follows. In Section II we propose modifications to TCP that will help the receiver in distinguishing corruption loss from congestion loss. We also show through graphs that there is clearly a high probability of the receiver identifying corruption loss correctly. In Section III we propose modifications to TCP sender behavior that utilize Explicit Loss Notification information. In Section IV we discuss the results and show that there is definite improvement in throughput for NewRenoEln as compared to NewReno. In the end we present conclusions and possible future work in this area.

II. Mechanism to Identify Corruption Losses

In this section we will explain the mechanism that we are proposing to distinguish between congestion and corruption loss. In our work we have assumed a system where there is

0-7803-7632-3/02/$17.00 ©2002 IEEE

a fixed node that is a sender and the receiver is connected to the network through a wireless link (Fig. 1). This wireless link is the main source of corruption errors for the packets transmitted by the sender. For the sake of analysis we will also assume that congestion losses are very few and can be ignored as compared to corruption losses.

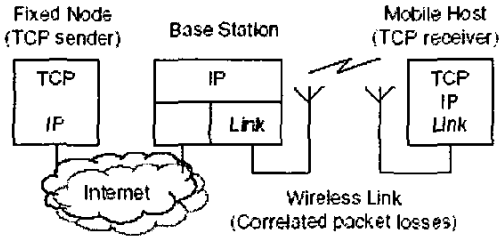

Fig. 1. Mobile Host connected to Internet over Wireless link.

We have assumed a generic link layer protocol between base station and mobile host. The base station will fragment the TCP packet into multiple frames and will add some error control bits before transmitting the packet on the wireless link. Due to the high bit error rate on the wireless link, some of these frames will be lost or will be received in error by the receiver. Normally, the link layer protocol will discard such corrupted frames and the TCP/IP layer will not receive the complete packet. We propose that link layer protocol be modified such that it does not discard corrupted frames. TCP/IP layer will anyway calculate the checksum on the packet and will detect the error. Since the checksum is calculated over both the header and data portions, the receiver cannot believe the header information of the corrupted packet, as the header itself might have got corrupted. The TCP protocol at the receiver discards such packets and no information is sent to the sender.

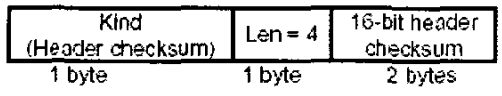

Fig. 2. TCP Header Checksum Option, along with data packets from sender to receiver.

We propose that the TCP header along with the pseudo-header be separately protected by their own header checksum. The TCP header is 20 bytes long and consists of source and destination port number, sequence number and acknowledgment number, header length, control flags, window size, TCP checksum and urgent pointer. The pseudo-header as defined in [1] includes source and destination IP address, protocol field and 16-bit total length of packet. The purpose is to let TCP double-check that the data has arrived at the correct destination (i.e., that IP has not accepted a datagram that is not addressed to this host, and that IP has not given TCP a datagram that is for another upper layer e.g. UDP). The header checksum will be calculated on all the fields of TCP header and pseudo-header except the TCP checksum (original checksum that is calculated for both header and data). This header checksum will be carried in the option field (Fig. 2) that will be attached to TCP header.

During connection setup, sender and receiver will exchange information on whether to use this new scheme or not, using the SYN packets. While requesting for a connection, the receiver (mobile host) will add an option field (Fig. 3) to the SYN packet asking for permission to send ELN based on TCP header checksum along with ACK packets. The sender (fixed node) will permit the use of this option using the same option field along with the SYN packet in reverse direction.

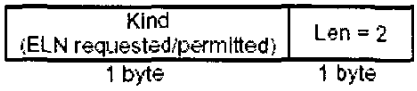

Fig. 3. Explicit Loss Notification requested Option, along with SYN packet from receiver to sender and ELN permitted option, along with SYN packet from sender to receiver.

This option will not be sent in non-SYN packets. If the sender does not receive this option in the SYN segment then it must not send header checksum on the data packets. If the receiver does not receive this option in the SYN segment then it must not generate ACK packets with ELN information.

The TCP/IP protocol is modified in the receiver to check the checksum on the complete TCP packet that includes header and data, and to separately calculate the checksum on the header portion. In case the checksum check on the whole packet fails but the check on the header is successful, then surely this is a case of corruption of packet on the wireless link. In this case the receiver will generate reliable ELN for the sender, as it now has the complete socket address information as well as the sequence number of the lost packet. In case the checksum check fails for the header, the packet is silently discarded.

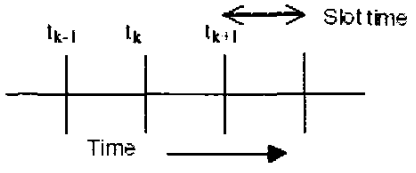

Fig. 4. Slotted timeline for packets/frames over wireless link.

Now we will evaluate the probability that a receiver is able to provide ELN to the sender. We will follow the analysis for packet loss in wireless link with correlated losses in [11]. We model the wireless link as a discrete time Markov chain with two states: good and bad. Time is slotted in units of packet/ frame transmission time (Fig. 4). The analysis will use both frame and packet transmission times.

We assume that packet transmission starts only at slot boundaries t_k, t_{k+1} etc. If the channel is in the good state at time t_k^-, then the transmission starting at time t_k will be

successful with probability 1 and when the channel is in the bad state at time t_k^-, then the packet/frame transmission starting at time t_k will fail with probability 1. Also it is assumed that the channel state changes only at the slot boundary. For such a channel the transition probability matrix is given by

$$M_{CP}(x) = \begin{pmatrix} P_{BB}(x) & P_{BG}(x) \\ P_{GB}(x) & P_{GG}(x) \end{pmatrix} \quad (1)$$

$$M_{CP} = \begin{pmatrix} P_{BB} & P_{BG} \\ P_{GB} & P_{GG} \end{pmatrix}, \quad \text{where} \quad M_{CP}(x) = (M_{CP})^x \quad (2)$$

Here $M_{CP}(x)$ denotes the x-step transition probability matrix and M_{CP} denotes the single-step transition probability matrix. Thus $P_{GG}(x)$ denotes the probability that transmission in slot i is successful given that the transmission in slot $i-x$ was successful. Similarly P_{GB} denotes the success to failure probability, P_{BG} denotes the failure to success probability and P_{BB} denotes the failure to failure probability.

Given the matrix M_{CP}, the channel properties are completely characterized. In particular, it is possible to find the steady state distribution of the chain. The steady state probability that a packet error occurs, P_E, is

$$P_E = \frac{1 - P_{GG}}{2 - P_{GG} - P_{BB}} \quad (3)$$

Also, for a Rayleigh fading channel with a fading margin F, the average packet error rate can be found as

$$P_E = 1 - e^{-1/F} \quad (4)$$

The Markov parameter P_{BB} is given as

$$P_{BB} = 1 - \left(\frac{Q(\theta, \rho\theta) - Q(\rho\theta, \theta)}{e^{1/F} - 1} \right), \quad \text{where} \quad \theta = \sqrt{\frac{2/F}{1-\rho^2}} \quad \text{and} \quad \rho = J_0(2\pi f_d T) \quad (5)$$

ρ is the Gaussian correlation coefficient of two successive samples of the complex amplitude of a fading channel with Doppler frequency in Hertz f_d, taken T seconds apart. f_dT is the normalized Doppler bandwidth and is the indication of correlation in the wireless channel. Lower value of f_dT (= 0.01) indicates high correlation in the channel and larger values (= 0.5) indicates low correlation approaching (Independent Identically Distributed) IID characteristics for the channel. f_d is calculated as

$$f_d = \frac{v f_c}{c} \quad (6)$$

where v is the velocity of the mobile host in meters/sec, f_c is the carrier frequency in Hertz and c is the speed of light in meters/sec. T is the packet transmission time in seconds on the channel and is calculated as

$$T = \frac{K_P}{bw} \quad (7)$$

where K_P is the packet size in bits and bw is the bandwidth of the channel in bits/sec. $J_0(.)$ is the Bessel function of the first kind and zero order. $Q(.,.)$ is the Marcum Q function given by

$$Q(x,y) = \int_y^\infty e^{\frac{-(x^2+w^2)}{2}} I_0(xw) w dw \quad (8)$$

$I_0(.)$ is the modified Bessel function of first kind and zero order. Now we can calculate the channel parameters for transmission of TCP packets of different sizes and for different values of packet error probability, speed of mobile and bandwidth available for transmission.

TABLE I
NORMALIZED DOPPLER BANDWIDTH f_dT FOR VARIOUS VALUES OF MOBILE SPEED AND BANDWIDTH OF WIRELESS LINK.

S. No.	Mobile Speed in Km/hr	Bandwidth of wireless link	f_dT
1.	1.8	100Kbps	0.168
2.		1Mbps	0.0168
3.		2Mbps	0.0084
4.	36	100Kbps	3.36
5.		1Mbps	0.336
6.		2Mbps	0.168
7.	90	100Kbps	8.4
8.		1Mbps	0.84
9.		2Mbps	0.42

We can easily adapt the same approach to the frames. The slot time is defined as the frame transmission time in this case. Here we can calculate the Markov channel parameters for a given F_E (Frame error probability), mobile speed, bandwidth and size of frame K_F. Here

$$K_F = \frac{K_P}{N} \quad (9)$$

when a packet of size K_P is fragmented into N (fragmentation factor) frames, each of size K_F. We can denote the transition probability matrix for frames as

$$M_{CF}(x) = \begin{pmatrix} F_{BB}(x) & F_{BG}(x) \\ F_{GB}(x) & F_{GG}(x) \end{pmatrix} \quad (10)$$

The steady state probability that a frame error occurs is given by

$$F_E = \frac{1 - F_{GG}}{2 - F_{GG} - F_{BB}} \quad (11)$$

We will now calculate packet error probability given the transition probability matrix for the frames. A packet will be received in error if any of the frames is received in error. This can be written as

$$P_E = 1 - \left[(1 - F_E)\ (F_{GG}^N) + F_E F_{BG} F_{GG}^{(N-1)} \right] \quad (12)$$

Here we are first calculating the probability that all the N frames of the packet are received without error, when the

previous frame was received in either good or bad condition, and then subtracting it from one. In our analysis we assume that the receiver will be able to correctly decode header information only when the first and last frames of the packet are received correctly. Thus, if the first and/or last frame of the packet is found in error, we will conclude that header is found in error. We will denote this term by header error probability H_E. This is given by

$$H_E = ((1 - F_E)\, F_{GB} + F_E F_{BB}) + ((1 - F_E) F_{GG} + F_E F_{BG})\,(F_{GB}(N - 1)) \quad (13)$$

In the above expression the first term denotes the case where the first frame is received in error. The second term denotes the case where the first frame is received successfully but the last frame is received in error. Let us denote the probability that the receiver will send successful ELN information to the sender given that packet is received in error by P_{ELNS}. Also P_{ELNF} denotes that receiver will not send ELN information to the sender. Now P_{ELNF} is just the probability that first or last frame is received in error given that packet was received in error. This is given by

$$P_{ELNF} = \frac{H_E}{P_E} \quad \text{and} \quad P_{ELNS} = 1 - \frac{H_E}{P_E} \quad (14)$$

Using the expressions we have derived above, we can now calculate the probability that the sender will receive ELN information from the receiver, given the packet error rate. We assume that there are no errors in the feedback path.

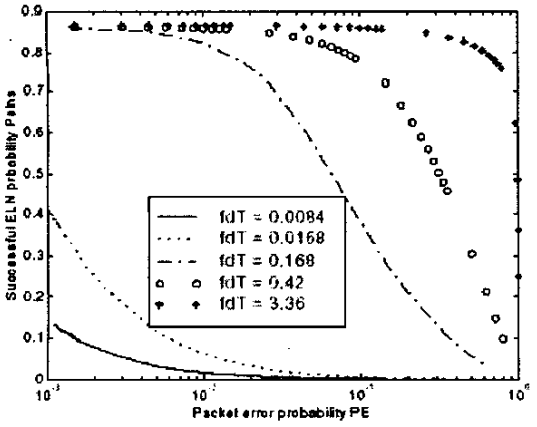

Fig. 5. Successful Explicit Loss Notification Probability versus Packet error probability, for $N = 15$ and different values of f_dT.

There is high probability that the receiver will be able to distinguish corruption loss from congestion loss (Fig. 5), when the channel is not highly correlated (for high values of f_dT). In case the channel is highly correlated (for low values of f_dT), then it is more probable that either the first or last frame of the packet will be in error and thus the receiver will not be able to distinguish congestion loss from corruption loss.

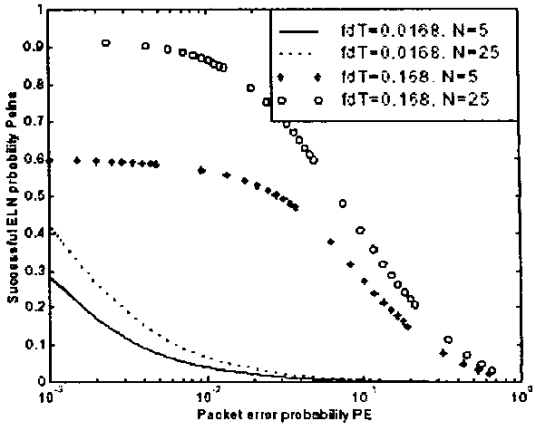

Fig. 6. Successful Explicit Loss Notification Probability versus Packet error probability, for different values of fragmentation factor N.

Fig. 6 shows that for a high fragmentation factor there is more chance that the receiver will receive the first and last frames without any error.

III. TCP Recovery for NewRenoEln

In this section we will explain the recovery procedure that sender and receiver should undertake once a corruption loss is identified. If the receiver finds that the received packet is corrupted but the header checksum validates that header information is correct, then the receiver will generate the duplicate ACK and it will also attach ELN ACK option field to the packet.

Kind (ELN ACK)	Len = 6	32-bit sequence number of corrupted segment
1 byte	1 byte	4 bytes

Fig. 7. ELN ACK option along with duplicate ACK packets

When the sender receives an ACK packet without the ELN ACK option field, it will process it according to NewReno protocol. If the sender receives a duplicate ACK with ELN ACK option attached, then it will retransmit the corrupted packet, as indicated by the ELN ACK option field. It will also restart the timeout timer for this packet. The sender will not make any changes to the congestion window parameters.

IV. Simulation Results and Discussion

We have simulated NewRenoEln TCP protocol using Berkley's network simulator ns [12] for various conditions including different mobile host speeds, bandwidth of wireless link and fragmetation factor. We assume a correlated channel and at the receiver we take the P_{ELNS} values as derived in the Section II to generate duplicate ACK with ELN ACK option field. We have compared the throughput of NewRenoEln with the throughput achieved by NewReno protocol in similar conditions. Some of these results are presented here. In all the graphs we assume dupack threshold

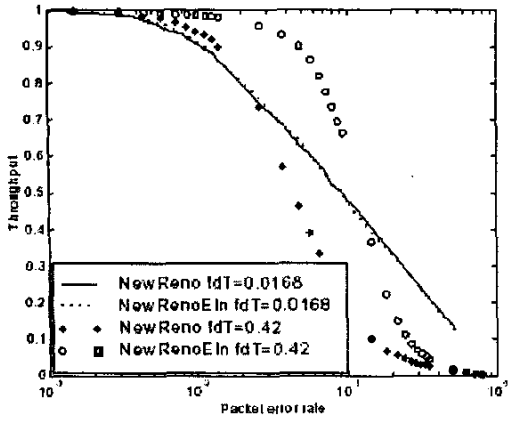

Fig. 8. Throughput performance of TCP NewReno and NewRenoEln versus packet error rate. $W_{max} = 6$, $N = 15$.

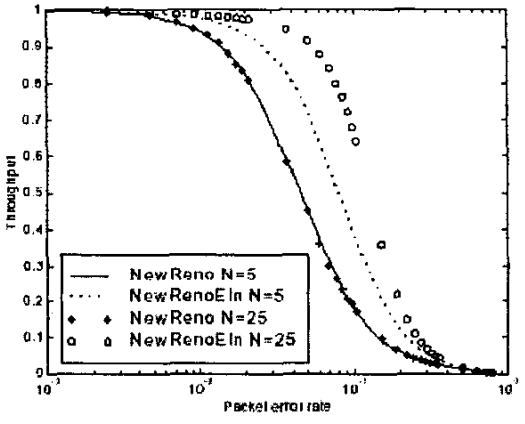

Fig. 9. Throughput performance of TCP NewReno and NewRenoEln versus packet error rate. $W_{max} = 6$, $f_dT = 0.42$.

$K = 3$, timeout is fixed and is equal to 100 packet transmission times on the wireless link. The TCP packet size is fixed at 1400 bytes. The sender always has data to send, and the user at the mobile host receives the data as soon as the TCP receiver provides it. We also assume that the sender receives ACKs instantaneously and without any loss.

For very high correlation in the channel ($f_dT = 0.0168$), NewRenoEln performs no better than NewReno protocol (Fig. 8). With high correlation in the channel there is high probability that either the first or last frame of the packet will be corrupted and the receiver will not be able to identify the loss as being caused by corruption. When the correlation in the channel is low ($f_dT = 0.42$), NewRenoEln performs substantially better than NewReno.

The NewRenoEln protocol will also perform better with high fragmentation factor N (See Fig. 9). When N is high there is a large probability that the frames containing header information will be correctly received by the mobile host.

V. Conclusions and Future Work

We have shown in this paper that using a separate checksum for the TCP header provides mechanism for differentiating congestion loss from corruption loss. We have developed the analysis for a generic link layer protocol in this paper. We have also shown that the NewRenoEln protocol, based on successful detection of corruption losses, will perform better than NewReno under various conditions. We have shown the effect of correlation in the channel and fragmentation factor on the performance of NewReno and NewRenoEln protocol.

There are various possibilities for future work. This includes extending the generic link layer protocol to link layer protocol of specific technology (e.g. Radio Link Protocol RLP of GSM or IEEE 802.11 WLAN) and studying the effect of using header checksum. We also need to see the behavior of the protocol in the actual network environment.

Acknowledgment

The authors would like to thank Mr. Satish Kulkarni and many others from C-DOT and CEDT for their support in this work.

References

[1] W. R. Stevens, *TCP/IP Illustrated, Volume 1*, Reading, MA: Addison Wesley, 1994.
[2] J. Postel, "Transmission Control Protocol," RFC793.
[3] H. Balakrishnan, V. N. Padmanabhan, S. Seshan and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *ACM/IEEE Trans. Networking*, vol. 56, pp. 756-759, Dec. 1997.
[4] T. Goff, J. Moronski, D. S. Phatak and V. Gupta, "Freeze TCP a true end to end TCP enhancement mechanism for mobile environments," in *Proc. IEEE INFOCOM*, 2000.
[5] R. Caceres and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE J. Select. Areas Commun.*, vol. 13, pp. 850-857, June 1995.
[6] H. Balakrishnan and R. H. Katz, "Explicit Loss Notification and Wireless Web Performance," in *Proc. IEEE Globecom Internet Mini-Conference*, Nov. 1998.
[7] F. Peng and J. Ma, "A proposal to apply ECN into Wireless and Mobile Networks," draft-fpeng-ecn-03.txt.
[8] N. Vaidya, "Discriminating congestion losses from wireless losses using inter arrival times at the receiver," in *Proc. IEEE ASSET*, 1997.
[9] C. Parsa, J. J. G. L. Aceves, "Differentiating congestion vs. random loss: A method for improving TCP performance over wireless links," in *Proc. IEEE WCNC* 2000.
[10] R. K. Balan, B. P. Lee, K. R. R. Kumar, L. Jacob, W. K. G. Seah, A. L. Ananda, "TCP HACK: TCP header checksum option to improve performance over lossy links," in *Proc. IEEE INFOCOM*, Apr. 2001.
[11] M. Zorzi, R. R. Rao and L. B. Milstein, "On the accuracy of a first-order Markov model for data transmission on fading channels," in *Proc. IEEE ICUPC'95*, Nov. 1995, pp. 211-215.
[12] UCB/LBNL/VINT Network Simulator – ns (version 2). URL: http://www.isi.edu/nsnam/ns.
[13] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC2582.
[14] A. Kumar, "Comparative performance analysis of versions of TCP in a local network with a lossy link," ACM/IEEE Trans. Networking, pp. 485-498, Aug. 1998.