A link layer protocol for improving TCP performance over Satellite Channels

Ketan Bajaj
ketan@iitd.ernet.in
Huzur Saran
Deptt. of Computer Science & Engg.
Indian Institute of Technology,
New Delhi-110 016, India

Abstract

In this paper we look into the problems with transmission control protocol's fast retransmission and timeout [9] procedures in networks with satellite links, and propose a link layer solution. The satellite link layer protocol called Sat-LL which is designed here makes the losses over the satellite link transparent to the connection end points [figure 1]. Sat-LL does this by preventing the traversal of unnecessary duplicate acknowledgements which are generated due to satellite link losses. Besides this, we propose additional methods to improve both throughput and goodput over the satellite link. These include prevention of unnecessary retransmissions over the satellite link and explicit feedback messages to the data sender.

1 Introduction

In this section the reasons for TCP's poor performance over satellite channels are given. In section 2, we describe the protocol which solves this problem and in section 3 we conclude with an analysis of the design.

Current TCP implementations such as TCP Tahoe and TCP Reno [4, 9] undergo fast retransmission on receiving three duplicate acknowledgements. This helps in an early recovery from congestion losses as compared to the recovery done after a retransmission timer expires. In TCP Reno the output window is reduced to half of its current size, whereas in TCP Tahoe the slow start procedure is initiated, reducing the output window to one MSS[Maximum Segement Size]. This helps in preventing further congestion losses.

Though fast retransmission works well for networks with only congestion losses it leads to poor performance in networks with satellite links. Satellite links exhibit losses due to packet corruption. For any such loss duplicate acknowledgements are generated by the receiver TCP. On receiving three such duplicate acks the sender TCP reduces its window leading to an unnecessary reduction of throughput.

Besides fast retransmission, the TCP timeout procedure also gives poor performance for satellite channels. Link layer retransmissions help in recovering the losses over the satellite link, but as the delay over a satellite channel is large the recovery time is also high. This gives sufficient time for the sender TCP to timeout and reduce the throughput unnecessarily.

The fast retransmissions and timeouts due to satellite link losses besides leading to a decrease of throughput also cause unnecessary retransmissions, leading to a reduction of goodput at the receiver. This also leads to redundant retransmissions over the satellite channel.

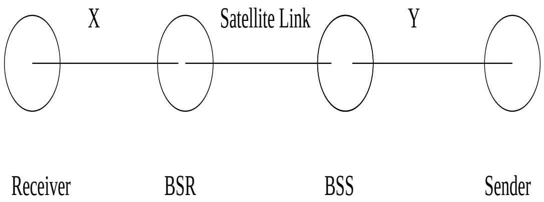

Figure 1: A simple satellite topology

Current work on improving performance of TCP over satellite channels and the research issues involved are summarised in [6] & [7] respectively. Wireless LANs share some characteristics with satellite channels as both incur data loss due to reasons other than congestion. Protocols for improving TCP performance in mobile networks have been developed in [1, 3, 8].

2 Protocol Description

The protocol requires modifications to the base stations at the end points of the satellite link. These are BSS (Base Station towards the data sender end) and BSR (Base Station towards the data receiver end), see figure 1. Paths 'X' & 'Y' may constitute a number of intermediate terrestial/satellite hops. The base stations are made capable of looking into the transport layer packet sequence numbers. A similar concept was used in [3] for a single base station in a wireless LAN. The data, ack and message events occuring in a single round trip are explained in the following sections, omitting detail about message formats.

2.1 BSS Data Behaviour

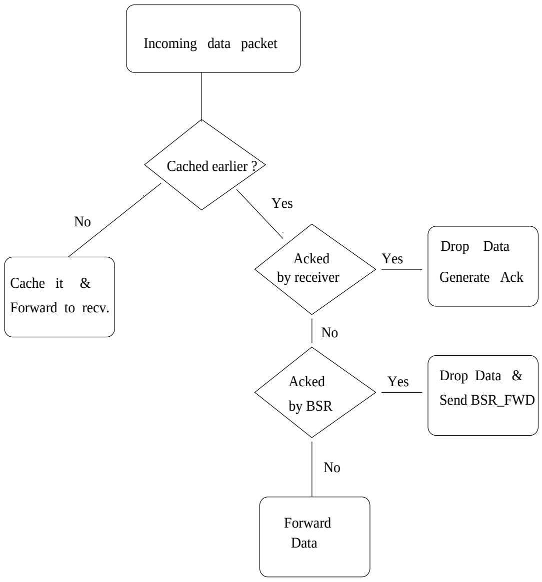

Figure 2: Data events at BSS

The BSS maintains a cache of all packets which have not been received before. All such new packets which may arrive in sequence or out of sequence at BSS are cached and forwarded to the receiver.

For the packets which have been cached earlier, BSS checks whether the packet is acknowledged by the receiver. If it is then this particular packet is dropped thus preventing an unnecessary retransmission over the satellite link. An ack is generated for the sender. If not acked by receiver then BSS checks whether the packet is acked by the other end of the satellite link i.e. the BSR. If it is, then BSS drops the packet and sends a BSR-FWD message to BSR. The BSR-FWD message maintains the TCP header but instead of sending the complete data it contains the data beginning and ending sequence numbers. Reducing the size of retransmitted data over the satellite link reduces link layer fragmentation and also reduces the transmission time. If it is acked neither by receiver or BSR, then it is forwarded as such to receiver.

2.2 BSR Data Behaviour

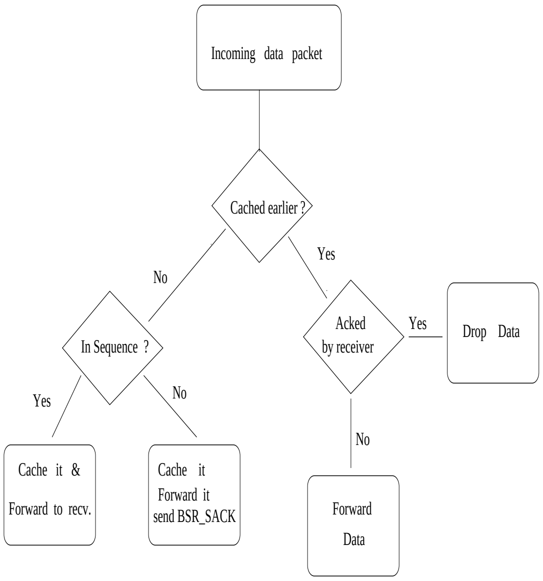

Figure 3: Data events at BSR

BSR in the normal case, on receiving a new packet, whether in sequence or out of sequence, caches it and forwards it to the receiver. For packets arriving at BSR which have been cached earlier, BSR checks if the packet is acked by the receiver. Packets acked by the receiver are dropped thus preventing unnecessary retransmissions to go over the BSR to receiver path. Acknowlegements are generated for such packets. Retransmissions arriving at BSR which are not acked by receiver are forwarded as such.

On receiving a BSR-FWD message BSR ignores the message if the data to be forwarded is already acked by the receiver and generates an ack. Otherwise, it forwards the data and drops the message. BSR-FWD messages remain transparent to the connection end points.

The working of Sat-LL remains independent of the recovery mechanism used over the satellite link. In our experiments BSS maintains an estimate of the round trip time over the satellite link and does a retransmission when a timer expires. Timeout events at BSS are cancelled on receiving link layer acks from BSR.

2.3 BSR Ack Behaviour

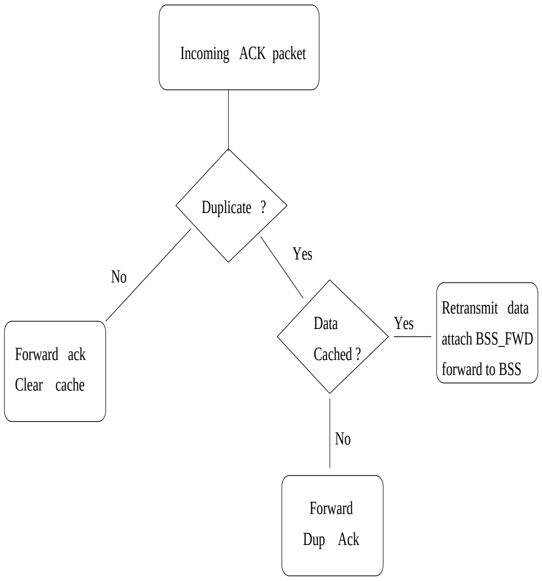

Figure 4: Ack events at BSR

An ack from receiver acknowledges some cached data at BSR. This is a normal case. BSR clears its cache and forwards the ack to the sender.

If the BSR receives a duplicate ack, i.e. the sequence number in the incoming ack acknowledges no new data in the cache, then it implies a gap in the receiver's queue. If data is cached at BSR then it is a loss over the BSR-receiver (X) path due to congestion. So, BSR retransmits the data to the data receiver on receiving three such duplicate acks. Though this is not a necessary part of the protocol, it helps in an early recovery of congestion losses over the BSR-receiver path. The sender should be informed that that a congestion loss has occured, so that it implements its usual TCP congestion control mechanism. So, BSR forwards the duplicate ack but attaches a BSS-FWD header to it so that the BSS does not drop this duplicate ack [section 2.4]. The BSS-FWD can be implemented as a single bit option in the TCP header, which is removed by BSS. For a satellite link loss, duplicate acks will be generated for each packet reaching the receiver until the retransmission from BSS reaches the receiver. When the retransmission reaches BSR, it may trigger a false BSS-FWD message. The implementation of Sat-LL takes care of this, and no BSS-FWD's are attached for such duplicate acks.

If the BSR receives a duplicate ack and there's gap in its cache starting from the sequence number in the duplicate ack, then it implies a loss of data over the sender-BSR path or misordering over the sender-BSS path. In this case BSR forwards the duplicate ack to BSS, which takes the appropriate action, whether to drop or forward it [section 2.4]. Here, BSR attaches no BSS-FWD header to the duplicate ack.

2.4 BSS Ack Behaviour

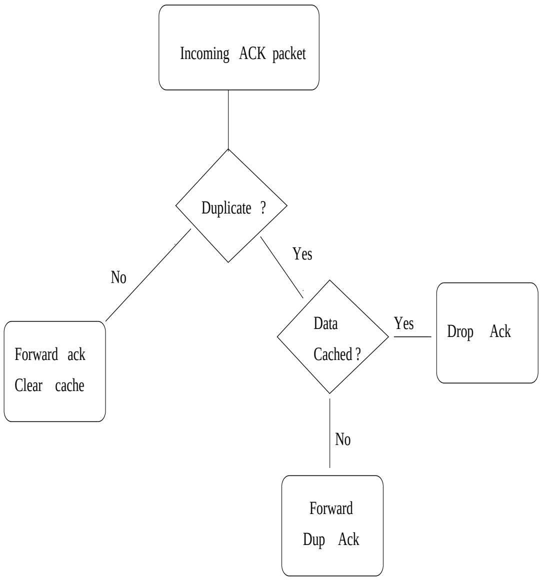

Figure 5: Ack events at BSS

During normal conditions when BSS receives an ack from receiver which acks some cached data at BSS, the cache is cleared and the ack forwarded to the sender.

If the ack is a BSS-FWD message created by BSR, then BSS removes the BSS-FWD option from the TCP header and forwards the packet as such. This is because this duplicate ack is due to congestion loss over BSR-receiver path, and the sender must be informed of it.

If the ack from receiver is a duplicate ack then BSS checks its cache. If the data packet is present then BSS deduces a loss over the satellite link and drops the duplicate ack. Otherwise if the data packet isn't present BSS concludes a loss over the sender-BSS path and lets the duplicate ack go to the sender. This method of dropping duplicate acks which arise due to losses over the satellite link makes the satellite link losses transparent to the sender. Thus, preventing unnecessary fast retransmissions by sender TCP.

Though local retransmissions help in recovering the losses over the satellite link, this does not prevent the sender TCP from timing out. The time it takes for a retransmission from BSS to reach the receiver and the acknowledgement to traverse back gives sufficient time for the sender to timeout and enter Slow Start. To prevent these unnecessary timeouts at sender, BSS sends an explicit feedback message(EFM) to sender every time it retransmits on a time out. The sender's response to EFMs is given in section 2.5. To make the EFM message transparent to the network, BSS creates an acknowledgement packet with an EFM option. The sequence number of the packet is the beginning of the gap at BSS's queue which is lost and the EFM option contains the ending sequence of this gap. The sender TCP on receiving such a packet extracts the sequence numbers and ignores the acknowledgement. The issues regarding EFM implementation have not been looked upon in detail yet, and could be inspired by the developments regarding Explicit Congestion Notification. [5]

2.5 Data Sender Behaviour

If EFMs are used, the data sender needs to be modified to process the EFM message and cancel the timeouts. On receiving an EFM message, the sender TCP can process it in two ways. It could cancel all timeout events corresponding to the sequence numbers extracted from EFM message, or it could let the timeout occur, but when such an event occurs change the way the congestion window (cwnd) and the slow start threshold (ssthresh) values are altered. For a timeout the ssthresh is set to half of the current TCP window size and the cwnd is set to one MSS (maximum segment size) [9]. This has been altered in Sat-LL in the following ways.

When the ssthresh and cwnd are kept same the loss is made completely transparent to the sender. If the cwnd is halved and ssthresh is set to the value of cwnd an attempt is made to reduce the drastic effect of timeouts by converting all timeouts to window halvings as in fast recovery [9]. Another method is to allow the timeout and set the congestion window to one MSS (maximum segment size). But, as there was a loss over the satellite link, so the ssthresh is not halved. This allows a quick recovery after the timeout as the TCP congestion window grows exponentially till the ssthresh value is reached. So, the TCP window achieves a higher value earlier. This has been called as Extended Slow Start. Figure 8 shows a trace of sender TCP's congestion window size when only satellite link losses occur and Sat-LL with EFMs is used. The ssthresh value remains at 40Kbytes as Extended Slow Start is used.

3 Analysis

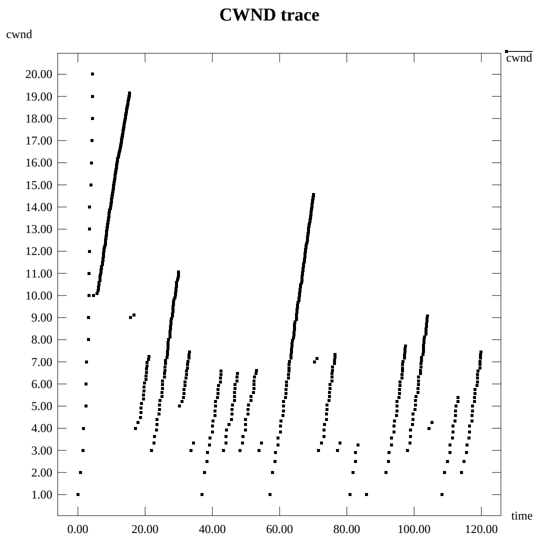

Figure 6: 8
Sender congestion window trace for only satellite link losses

In this section we analyse the Sat-LL protocol

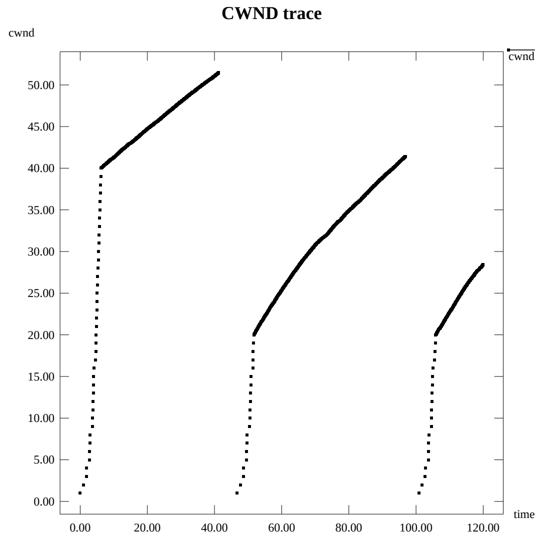

Figure 7: Sat-LL eliminating fast retransmissions

looking into the achievements obtained by this design.

Sat-LL is preventing all fast retransmissions which arise due to satellite channel losses, thus preventing an unnecessary reduction of throughput. Figures 6 and 7. show a trace of sender TCP's congestion window for no congestion losses in the network and data/ack packet loss occuring only over the satellite channel. Figure 6, which is without Sat-LL, shows multiple fast retransmissions and timeouts resulting in a poor performance. In figure 7, on using Sat-LL with BSS-FWDs, the fast retransmissions are eliminated and only timeouts are left. The TCP window size now achieves higher values (measured in Kbytes) leading to an imrovement of throughput.

Sat-LL takes all measures to preserve TCP congestion control semantics by using BSS-FWD messages which inform BSS to forward duplicate acks arising out of congestion losses over the BSR to receiver path.

If EFMs are used timeouts resulting from satellite losses can also be prevented or there effect mitigated by using Extended Slow Start as shown in [figure 8]. This leads to a further improvement in throughput.

Preventing timeouts and fast retransmissions prevents unnecessary retransmissions by the sender, thus leading to an increased goodput which is defined as the unique data per unit time arriving at the receiver.

BSS & BSR also prevent unnecessary retransmissions over the satellite channel and the BSR to receiver path respectively. This is because they do not send retransmitted packets which have already been acked by the receiver.

BSR-FWD messages reduce the size of retransmitted packets which have already been cached at BSR, thus reducing link layer fragmentation and the transmission time over the satellite channel.

Sat-LL maintains TCP end-to-end semantics which were violated in the split connection design given in [1], where the sender TCP could receive an ack before the packet actually reached the destination.

The design of Sat-LL is symmetric and works in the same manner for data transfer in either direction. In an actual implementation each of the ground(base) stations would have both the BSR and BSS capabilities.

If Sat-LL is used without retransmissions from BSR and without BSR-FWD messages then the additional cache requirements at BSR can be eliminated, and the protocol could be implemented by maintaining only the required variables at BSR and BSS.

The protocol when used without EFMs remains completely transparent to the hosts and the rest of the network. This transparency can facilitate an easier deployment in the existing satellite networks without causing any changes at the hosts.

A number of experiments were carried out using the network simulator NS2.1b3 from UCB & LBL. Different satellite networks topologies, congestion losses, error rates over the satellite channel were used. Figure 9. depicts a comparasion between Sat-LL and a vanilla satellite link, for only satellite channel losses with different exponentially distributed error means(in packets). Sat-LL shows an over 100% improvement in throughput. The throughput with Sat-LL reaches close to the network bottleneck of 128kbps.

The design of the Sat-LL protocol directly

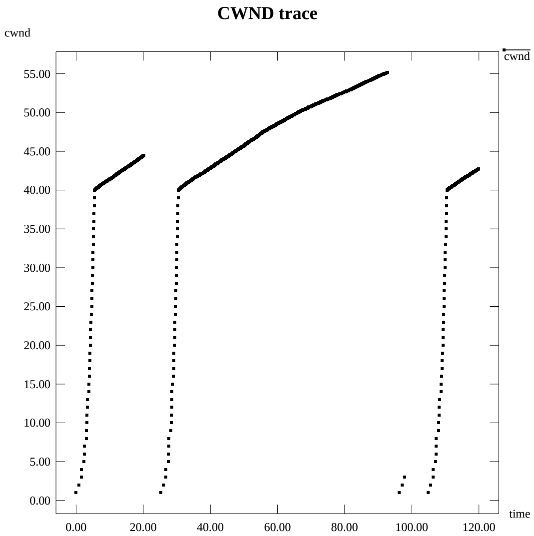

Figure 8: Sat-LL converting timeouts to Extended Slow Start

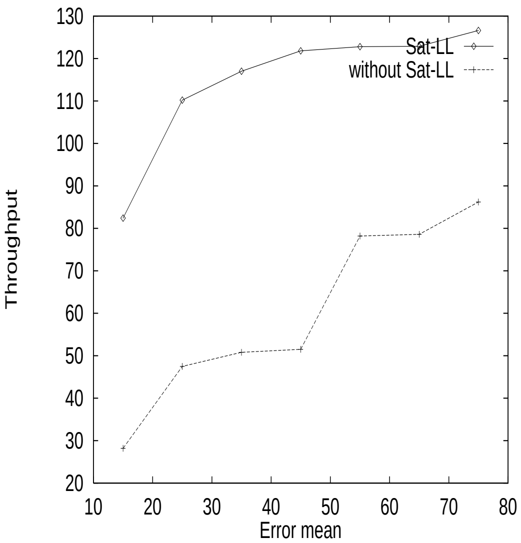

Figure 9: Throughput vs Error Mean for Sat-LL & without Sat-LL

addresses the problems(fast retransmission & timeouts) which TCP has over satellite channels, and introduces a new mechanism for improving TCP performance.

References

[1] A.Bakre and B.D.Badrinath*I-TCP:Indirect TCP for mobile hosts* Technical Report DCS-TR-314, Rutgers University(October 1994)

[2] A.DeSimone,M.C.Chuah and O.C.Yue *Throughput Performance of Transport-Layer Protocols over Wireless LANs*, in Proc.Globecom '93(December 1993)

[3] Hari Balakrishnan, Srinivasan Seshan, Elan Amir and Randy H.Katz.: *Improving TCP/IP Performance over Wireless Networks.*Computer Science Division. University of California at Berkeley.

[4] Kevin Fall and Sally Floyd: *Simulations-based comparasions of Tahoe, Reno and Sack TCP*, Lawrence Berkeley National Laboratory. One Cyclotron Road, Berkeley, CA 94370.

[5] K.K.Ramakrishnan AT&T Labs Research, Sally Floyd LBNL *A proposal to add Explicit Congestion Notification (ECN) to IP* Internet Engineering Task Force, Internet Draft, September 1998.

[6] Mark Allman,Dan Glover*Enhancing TCP over Satellite Channels using Standard Mechanisms* Internet Engineering Task Force, August,1998.

[7] Mark Allman, Spencer Dawkins, Dan Glover,Jim Griner,John Heidemann, Shaw Ostermann,Keith scott,Jeffrey Semke,Joe Touch,Diepchi Tran *Ongoing TCP Research Related to Satellite* Internet Engineering Task Force,August 1998.

[8] Ramon Caceres and Liviu Iftode: *Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments.*IEEE Journal on Selected areas in Communications. Vol 13.No 5.June 1995.

[9] W.Stevens: *TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms.* Network Working Group. RFC 2001. January 1997.