Enhancing TCP's performance over GEO satellite links with split-connections and link level retransmissions

M. Karaliopoulos[†] Rahim Tafazolli[†] Barry Evans[†]
Center for Communication Systems Research
University of Surrey
Guildford
Surrey GU2 7XH
(M. Karaliopoulos, R. Tafazolli, B. Evans)@eim.surrey.ac.uk

Abstract: The inefficiency of currently widely employed versions of TCP over satellite links has been reported extensively and several solutions have been suggested for the enhancement of its performance. Although the vast majority of them proposes modifications to the TCP implementation, it is accepted that TCP efficiency could be improved by a number of additional measures taken at other layers of the protocol stack: application-level solutions, employment of appropriate queuing disciplines and buffer management schemes, forward error correction and link-level interworking solutions are the most popular ones. This paper focuses on this last alternative; more specifically it explores by means of simulations the potential of split-connections coupled by link- level retransmissions to boost TCP's performance over GEO satellite links

1. Introduction

Future broadband satellite networks relying on GEO, MEO and LEO constellations intend to provide a wide variety of high and medium bit-rate data services to the end user. Given that IP applications in general and TCP/IP (WWW, e-mail, file transfer) more specifically are by far the most popular in the current data networks, it becomes quite evident that TCP will be the transport protocol of a significant part of the traffic of all networks, including satellite networks.

Quite a lot of research has been performed with regard to TCP performance in satellite environments, either in the frame of research teams [1][2][3] or separately by individual researchers[4][5], an extensive review of this being given in [1]. Although the majority of the solutions suggested concern the TCP implementation, It is accepted that TCP can be – and in some cases must be– helped by other layers, in order to overcome the limits that satellite environments pose to it. One of the cases where TCP seems to need the co-operation of other layers is the differentiation between congestion and corruption losses.

2. Split connections and link level retransmissions

TCP assumes that any packet loss (implicitly concluded either by means of duplicate acknowledgements or timeouts) is due to congestion, hence reduces its rate of transmission even when this is not appropriate. Some efforts to conclude between a corruption and a congestion loss relying on the statistics of the arrivals of the packets to the receiver or the Acknowledgements (ACKs) to the sender did not yield positive results [3]. Therefore, two seem to be the alternatives. Either TCP has to be explicitly notified about the reason of the packet loss (Explicit Congestion Notification [6] or Explicit Bad State Notification [7]) or the corruption losses must be hidden from TCP, so that it safely reacts in a congestion-oriented manner every time a packet is lost. An effective to way to achieve the latter is the employment of split connections coupled with link level retransmissions.

Split-TCP connections divide the connection at the gateway station, which is the border between the terrestrial and the satellite network. The TCP sender at the terrestrial

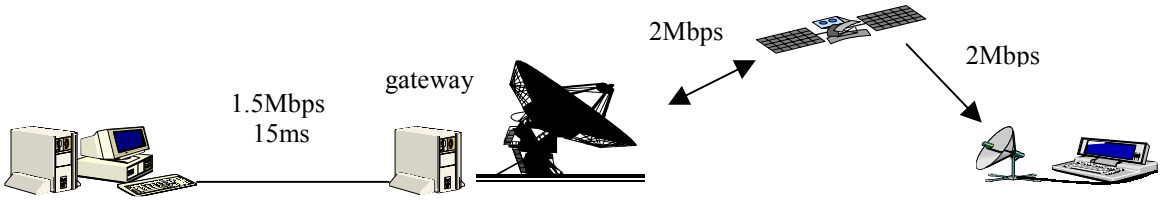

Fig. 1: Simulation scenario: a satellite host is connected via TCP with a server in the terrestrial network.

network does not have to be modified to respond to the challenges of a connection via satellite, while satellite specific solutions can be employed at the part of the connection that traverses the satellite link.

Link-level retransmissions are employed to cope with the packet losses due to errors at the satellite link. These can be data link layer retransmissions –an ARQ or hybrid FEC/ARQ system- or retransmissions made by a TCP-aware agent invoked at the satellite gateway. While a connection oriented data link layer protocol interacts badly with the TCP retransmission mechanism [8], the employment of less strict link layer protocol can yield significant performance improvements, as it will be shown below.

3. Simulation

3.1 Setup

The performance of five different TCP connections was examined in the scenario depicted in Fig.1: two end-to-end connections based on Reno (RENO E2E) and SACK (SACK E2E), two split connections , the one employing SACK TCP over the satellite link and the other Delayed Duplicate Acknowledgements (DelDupACKs) [9], and a replica of Snoop protocol [10] . Snoop and DelDupACKs -originally suggested for the terrestrial wireless networks- employ link level retransmissions.

The simulations were based on the abstract implementation of one-way TCP connections included in NS 2.1b6 [11]. Code for the abstract implementation of Snoop, DelDupACKs and split-connections over satellite was added to the simulator. The default value of 1000 bytes was adopted for the packet size. The sender's window was set to 32 while in the case of split connections the window of the TCP sender at the gateway was set to 100. The options for large initial window as well as timestamps were activated. The methodology for the comparison of the TCP versions was the one reported in [12]. Different amounts of data in the range from 4MB up to 12MB were transferred employing the 5 scenarios and the 'goodput' at the application level was estimated as the ratio of the total data transferred over the transfer time for different packet error rate (PER) values.

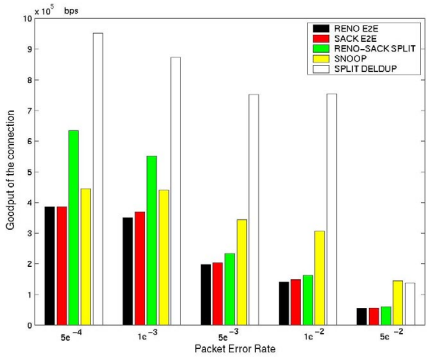

Fig. 2: Comparison of the performance of the five approaches under different values of the PER

3.2 Results-Comments

The relative (in)efficiency of the different approaches is clearly depicted in the representative diagram of figures 2. The end-to-end approaches exhibit the worst behaviour of all. Reno and SACK have practically the same performance since the uniform packet error model adopted at the satellite link prevents the latter from showing its enhanced ability to recover losses, in cases of multiple packet losses within the same window of data. The split-connection approach with TCP SACK at the satellite link, introduces a significant improvement at lower error rates, which tends to disappear as we move towards higher packet error rates. When the error rate is low,

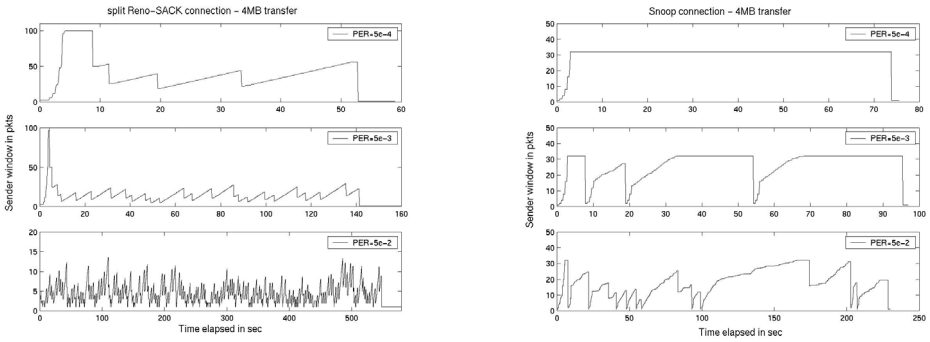

Fig. 3: The sender window of split Reno-SACK and snoop connections for different values of PER in the case of 4MB data transfer

the part of the connection over the satellite can exploit the larger TCP window that can be employed over the satellite link and increase its throughput. For higher error rates, the increased losses impose a much lower value on the sender's window cancelling this potential of split-connections (Fig. 3)

Snoop's performance at low error rates is limited by the terrestrial sender's window (maximum value=32 packets), but at higher error rates outperforms the split Reno-SACK connection. Introducing local retransmissions Snoop can isolate the sender from the losses encountered at the satellite link and prevent the sender from entering the Fast Recovery state. For higher error rates some timeouts at the sender cannot be avoided (Fig. 4)

The employment of the DelDupACKs in the frame of a split connection over the satellite link gives a significant improvement at lower error rates while at the extreme case of PER equal to $5e^{-2}$ the performance is slightly inferior to Snoop (fig .5). Since no congestion exists over the satellite link, the inefficiency reported in [8] with regard to the handling of congestion events is not a problem. The challenge in this case is an efficient choice of the delay of the third (or nth) duplicate ACK that will trigger Fast Recovery at the sender (a smaller value reduces the ability to perform local recovery of losses, while a higher one might cause senders timeouts).

4. Assumptions-Simplifications

The trade-off with regard to the *packet size* is evident. The bigger the packet, the smaller the transmission overhead and the higher the link utilization efficiency. On the other hand, smaller packet sizes mean smaller probability of packet corruption and reduced buffer requirements. Fragmentation can be performed either end-to-end employing Path MTU discovery or locally at the satellite links. The latter approach was shown to enhance performance in [10]. Fragmentation was not studied in this paper.

The error model of the satellite link is also a parameter that can vary the simulation

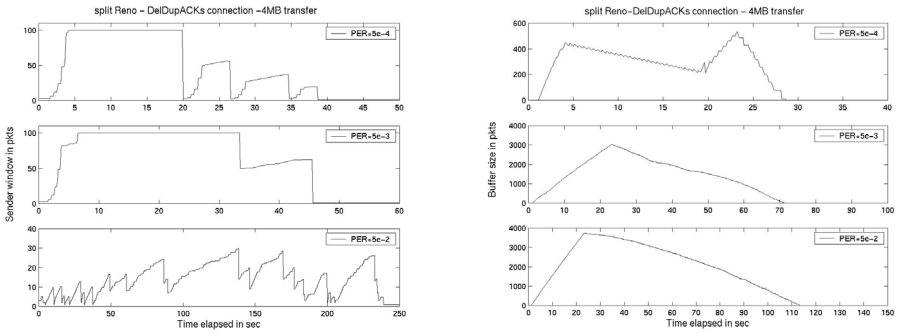

Fig. 4: Window variation and buffering requirements at the gateway for the split Reno-DeldupACKS connection case for different PER values. For high error rates, the requirements are almost equal to the size of transfer.

results. Different models may reveal different aspects of the algorithms behind the protocols and the same protocol can react in a different manner to different error models. The uniform packet error model was used in our simulations

5. Summary

TCP can benefit from the so called link layer interworking mechanisms to a significant extent. Split connections can improve performance, the penalty being the high buffering requirements at the gateway (Fig. 4) and the introduction of a single failure point at the network. SACK may be optimum in recovering losses but when the losses are due to transmission errors, we need a way to prevent TCP from taking congestion oriented action In other words we want a *preventive* approach rather than the *reactive* of SACK. Retransmissions –when performed locally- can prevent the fatal for the connection's throughput Fast Recovery triggering or timeouts. Delayed Duplicate Acknowledgement approach can be proven efficient every time there is packet loss and/or reordering not due to congestion and in our scenario this assumption holds. At moderate error rates DelDupACKs employed as the satellite part of a split connection shows improved performance over its competitors. Its inefficiency at the higher error rates necessitates a more systematic way of selecting the delay of the final duplicate ACK.

References

[1] M. Allman, S. Dawkins, D. Glover, J. Griner, T. Henderson, H. Kruse, S. Ostermann, K. Scott, J. Semke, J. Touch, D. Tran, 'Ongoing TCP Research Related to Satellites', informational, Internet RFC 2760, informational, October 1999

[2] R. C. Durst, G. J. Miller, E. J. Travis, 'TCP Extensions for Space Communications', Wireless Networks, pp. 389-403, October 1997

[3] S. Dawkins, G. Montenegro, M. Kojo, V. Magret, N. Vaidya, 'End-to end Performance Implications of Links with Errors', draft-ietf-pilc-error-03.txt, March 2000

[4] Partridge and T. Shepard, 'TCP/IP Performance over Satellite Links', IEEE Network, vol. 11, pp. 44-49, September/October 1997

[5] T. Henderson, R. Katz, 'Transport Protocols for Internet-Compatible Satellite Networks', IEEE Journal on Selected Areas in Communications, Vol. 17, No. 2, pp. 326-343, February 1999

[6] K. Ramakrishnan, S. Floyd, 'A Proposal to Add Explicit Congestion Notification (ECN) to IP', Internet RFC 2481, January 1999

[7] B. Bakshi, R. Krishna, N. H. Vaidya and D. K. Pradhan, 'Improving Performance of TCP over Wireless Networks', Technical Report 96-014, Texas A&M University, 1996

[8] N. Samaraweera, G. Fairhurst, 'Robust Data Link Protocols for Connectionless Service over Satellite Links', International Journal of Satellite Communications, vol. 14, pp. 427-437, 1996.

[9] N. Vaidya, M. Mehta, C. Perkins, G. Montenegro, 'Delayed Duplicate Acknowledgements: A TCP-Unaware Approach to Improve Performance of TCP over Wireless', Technical report, Texas A&M University, December 24, 1997

[10] H. Balakrishnan, S. Seshan, E. Amir, R. Katz, 'Improving TCP/IP Performance over Wireless Networks', Proceedings of 1st ACM International Conference on Mobile Computing and Networking (Mobicom), November 1995

[11] K. Fall, K. Varadhan, 'Ns Notes and Documentations'

[12] H. Balakrishnan, V. N. Padmanabhan, S. Seshan and R. H. Katz, 'A comparison of Mechanisms for Improving TCP performance over Wireless Links', Proc. SIGCOMM'96, August 1996.