Cross-Layer Enhancement to TCP Slow-Start over Geostationary Bandwidth on Demand Satellite Networks

Wei Koong Chai and George Pavlou

Centre for Communication Systems Research, University of Surrey, GU2 7XH, UK
{W.Chai, G.Pavlou}@surrey.ac.uk

Abstract. It is well-known that the transmission control protocol (TCP) does not perform well in wireless and satellite environments. We investigate the use of cross-layer design involving the transport and medium access control (MAC) layers in the context of a geostationary bandwidth-on-demand satellite network to simultaneously enhance TCP performance and to improve bandwidth utilization. In this paper, we focus on the slow-start phase of the connection. In essence, we create a bandwidth pipe between the two layers so that through cross-layer interactions, the TCP connections are aware of the satellite resources available to them, thus adjusting their congestion window accordingly. Our proposal includes minimal changes to the original protocol, allowing easier integration and inter-working with existing infrastructure. Our evaluation results show a shorter slow-start duration with better bandwidth utilization. Although the performance gain is higher in a lossy satellite link, we also found that it is dependent on the network load.

Keywords: Cross-layer design, satellite network, bandwidth on demand, TCP.

1 Introduction

The majority of Internet traffic uses the transmission control protocol (TCP). TCP was first conceived with terrestrial wired networks in mind, utilizing a closed-loop probing approach to slowly detect and utilize available resources in the end-to-end route. The protocol relies on feedback from the receiver in the form of ACK packets to gradually increase the sending rate. The classical TCP uses the additive increase, multiplicative decrease (AIMD) algorithm which increase the sending rate at a linear rate while decrease it exponentially in the event of losses which is assumed to be caused by congestion. In essence, it is a conservative protocol that is engineered to avoid severe Internet congestion.

A TCP connection consists of four phases: slow-start, congestion avoidance, fast retransmit and fast recovery [1]. Each connection starts with a slow-start phase and proceeds to the congestion avoidance phase. The fast retransmit and fast recovery phases are invoked to combat network problems such as TCP segment loss or data re-ordering. For long-lived TCP connections which mainly operate in the congestion avoidance phase, the effect of slow-start phase may be negligible. On the contrary, for short-lived connections, the TCP performance is highly dependent on the slow-start phase.

Although TCP performs satisfactorily in wired networks, the proliferation of satellite networks as an important component of the global information infrastructure presents specific challenges to TCP. The operating conditions and characteristics of satellite radio links are considerably different compared to wired links for which the TCP was first conceived. First, segment losses can no longer be assumed as a congestion indication since random losses caused by channel errors may not be negligible in satellite links. For instance, the channel quality of a *Ka band* satellite link is especially susceptible to atmospheric events such as rain. Second, the long propagation delay of satellite links has greatly lengthened the feedback process of the TCP protocol. Since TCP adjusts its sending rate per reception of ACK, the longer the round trip time (RTT), the slower it reacts to the current condition of the link. The problem is clear in a geostationary (GEO) satellite system which has approximately 560ms round-trip propagation delay, causing the TCP to increase its sending rate only once in more than half a second even in a lightly loaded link. The situation worsens when there are multiple packet losses. From a satellite operator's perspective, this is highly inefficient in minimizing bandwidth wastage. Satellite bandwidth is an expensive commodity which should be efficiently utilized. TCP should send data at the right rate so that the link is optimally utilized without causing congestion.

In this paper, we focus on the slow-start phase and propose a cross-layer mechanism between transport and medium access control (MAC) layers to enhance the TCP performance while improving bandwidth utilization in the context of a bandwidth-on-demand (BoD) GEO satellite system. Section 2 reviews the literature on TCP enhancements for satellite networks. We detail our reference system architecture in section 3. We elucidate our cross-layer enhancement in section 4. The performance of our proposal is then evaluated under different operating conditions via simulation and the results are presented in section 5. We summarize and conclude our work with discussions on its applicability and weaknesses in section 6.

2 Enhancing TCP for Satellite Networks

With the aforesaid problems, TCP has been recognized as inadequate for satellite networks. Over the last years, various proposals have been presented in attempting to solve these problems. In general, there exist three approaches. The first attempts to solve the problems by tuning specific TCP parameters without modifying the original TCP procedures. Since it involves minimal changes to the intrinsic working of the protocol, it is usually readily deployable. Its drawbacks are that the proposal usually targets a specific problem while ignoring others and that the improvement is limited. An example of this approach is [2] where a larger initial window is proposed. The second approach intervenes with the original AIMD mechanism of the protocol. TCP Peach [3], TCP Westwood [4] and TCP Hybla [5] are examples following this vein. These TCP variants often cope with various link characteristics but introduce integration and inter-working questions. The third approach isolates the satellite domain via third party proxies that split the TCP connections [6] and uses a different transport protocol specifically designed to suit the satellite environment within this domain. Interested readers are referred to [7][8] and the references therein.

Recent literature shows a new approach: cross-layer design which involves increased interactions between different layers within the protocol stack. The rationale behind this approach is that the current OSI model does not cater for sufficient adaptability to wireless satellite networks [9]. With the variability of the wireless link, it is beneficial to have the lower layers informing the upper ones the current link conditions so that the upper layers can adapt their behavior (e.g. sending rate) for an optimal operating equilibrium. For example, in [10], the cross-layer approach is employed for TCP and physical layer in assigning a common bandwidth resource to TCP connections over a satellite channel under different fading conditions. A TCP-MAC cross-layer resource allocation scheme is proposed in [11] to reduce the average file transfer time and to achieve a fair sharing of resources among competing flows. Ref. [12] suggests the use of cross-layer interactions between transport and MAC layers in a split-connection scenario where random error detection (RED) [13] is introduced at the MAC layer to provide congestion indication to TCP senders. The cross-layer approach is also utilized in [14] for enhancing the performance of TCP Westwood over satellite.

We refer to the ETSI BSM (Broadband Satellite Multimedia) protocol stack shown in Fig. 1 [15] which separates the network layers to satellite dependent (SD) layers and satellite independent layers (SI), connected via the Satellite Independent Service Access Point (SI-SAP). We exploit the fact that the SD layers can derive the exact available resources. Hence, if this information is communicated to the transport layer, the TCP sender will not have to be conservative in increasing its sending rate any longer. It can readily fill up the bandwidth without causing congestion.

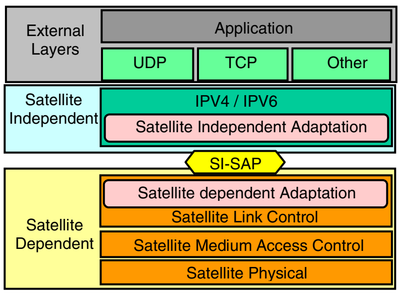

Fig. 1. Satellite BSM protocol stack [15]

3 Bandwidth-on-Demand Geostationary Satellite System

Our reference system, shown in Fig. 2, resembles the *Digital Video Broadcasting - Return Channel via Satellite* (DVB-RCS) architecture [16]. In view of next generation Internet, we assume the resource management scheduler is onboard the satellite. The network control center (NCC) provides control and monitoring functionality to the architecture. Users are represented via satellite terminal (ST). The traffic gateway (GW) provides connection to other domains (e.g. public and private providers). The satellite return link utilizes multi-frequency time division multiple access (MF-TDMA) and the basic capacity unit of an MF-TDMA frame is the timeslot (TS).

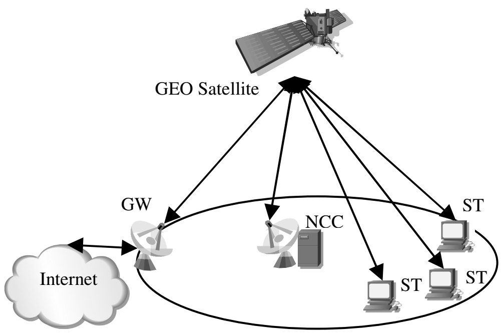

Fig. 2. Reference satellite network configuration

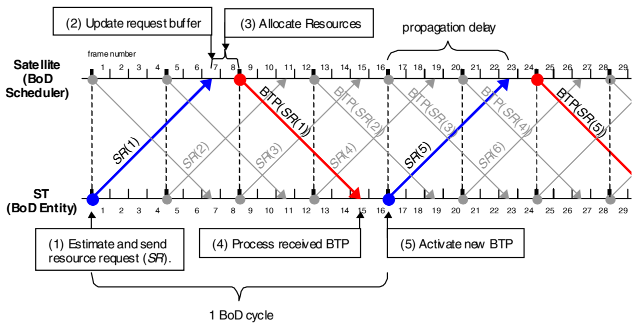

Fig. 3. BoD timing diagram

For efficient use of the satellite resources, a BoD scheme is specified. It is composed of two stages: the *resource request* from the STs and the *resource allocation* from the scheduler. Based on the incoming traffic, the STs estimate the resources required and then send a slot request (SR) to the scheduler. The computation of the SR is derived from [17]. The scheduler then allocates the TSs based on these requests. It constructs the burst time plan (BTP) that contains the allocation information and broadcasts it to all STs. Fig. 3 illustrates the evolution of the cyclic BoD process. We enable the *Free Capacity Assignment* (FCA) which will distribute the unassigned slots (after the allocation of all received SRs) to users so as to achieve higher efficiency in the system. The FCA basically reflects the spare capacity which can be utilized by the TCP connections.

4 Cross-Layer Enhancement for TCP Slow-Start Phase

4.1 Problem Statement

A TCP connection begins without the knowledge of available capacity and deploys the slow-start algorithm to probe the network with a small initial congestion window

($cwnd$) which governs the TCP send rate. The value of $cwnd$ is increased per reception of ACK by

$$cwnd = cwnd + 1$$

The connection exits the slow-start phase when the $cwnd$ exceeds the $ssthresh$ value which may initially be set as the advertised window. In a BoD satellite network, the MAC scheduler actually knows the exact availability of the bandwidth. Hence, by passing this information to the TCP sender, it can immediately increase the sending rate to fill up the available bandwidth without further probing.

Formally, if $cwnd_i$, $ssthresh_i$ and $cwnd_i^{incr}$ are the current $cwnd$, current $ssthresh$ and allowable $cwnd$ increase for connection i respectively, where $i \in I \equiv \{1,2,3,\ldots,N\}$ and N is the number of connections in the network, we formulate the problem as

$$Max(Z) = \sum_{i=1}^{N} cwnd_i^{incr} \tag{1}$$

subject to the constraints:

1. $\sum_{i=1}^{N}\left(cwnd_i + cwnd_i^{incr}\right) \le C_{total}$
2. $cwnd_i \le ssthresh_i; \quad \forall i \in \{1,\ldots,N\}$
3. $cwnd_i^{incr} \le TS_i^{FCA}; \quad \forall i \in \{1,\ldots,N\}$
4. $\dfrac{cwnd_i^{incr}}{cwnd_j^{incr}} \approx 1; \quad \forall i,j \in \{1,\ldots,N\}$

The first constraint ensures that the total sending rate after the increment of $cwnd$ does not exceed the total capacity, C_{total}, in the network. Since we focus only on the slow-start phase of the connection, the second constraint is included. The third constraint is required as the extra $cwnd$ increment of each connection cannot exceed the slots allocated by the FCA, TS_i^{FCA} to connection i. The fourth constraint is important to achieve fairness for all connections.

4.2 Algorithm of Cross-Layer Enhanced Slow-Start Phase

There are several design issues to be addressed. The foremost being the need for a cross-layer interaction facility interfacing between transport and MAC layers. We use a simple cross-layer entity that acts as an intermediary between them. The second design issue is the timescale separation between the two layers. The TCP sender adjusts its $cwnd$ every RTT which is more than half a second for satellite networks. However, the *system response time* for the BoD which accounts for the time between the sending of an SR to the activation of the corresponding BTP is much less (96ms for our case). Our solution operates on the transport layer timescale since the update from the BoD is more frequent so that TCP sender may have more up-to-date information to act up on. Another issue is on how the BoD should distribute its free slots.

Conventionally, this is done based on the STs whereby the free slots are allocated to all connected STs in a round-robin fashion. Since all STs will receive equal amount of free slots, it is unfair to those TCP connections residing in an ST that has high number of connections. Alternatively, the free slot allocation can be done based on TCP connections. This will ensure fairness to all connections but will burden the satellite scheduler since it has to keep track of all the TCP connections within the network. We get around this by assuming each ST has the same number of TCP connections. Although unrealistic, our aim is to understand the benefits and effectiveness of the mechanism rather than solving the details of its deployment. Satellite operators can decide on this issue based on their own policies.

In addition to that, we also exercise caution in designing our mechanism in order to avoid "spaghetti design" that consists of many convoluted cross-layer interactions. Voice for caution in cross-layer design has already been raised in [18]. Hence, it is still important that a certain level of modularity and integrity of the entire protocol stack are maintained. We cannot sacrifice long term performance guarantees with some immediate short term improvement to a certain performance metric.

Although the TCP and BoD module lie in non-adjacent layers, they are in fact inter-dependent. The sending rate of the TCP affects the resource allocation in the BoD scheduler while the BoD resource allocation impacts the RTT parameter for TCP. Our solution involves both horizontal (i.e. between MACs of STs and the satellite) and vertical (i.e. between transport and MAC layer within each ST) interaction implementations. The horizontal plane implementation ensures that the scheduler onboard the satellite allocate the requested slots appropriately together with fair allocation of spare capacity to all STs while the cross-layer communication is implemented via the vertical plane. Table 1 and Table 2 show the pseudo code for the STs and satellite BoD scheduler respectively.

Table 1. Pseudo code for ST involving both MAC and Transport layers

```
1.   MAC::BEGIN(t = request time)
2.   If (new BTP = TRUE)
3.   {    read BTP;
4.        compute TSs allocated by request;
5.           match MAC frame to the TSs for transmission;
6.        compute TSs allocated by FCA;
7.           if (TSs allocated by FCA > 0)
8.           {       translate TSs to cwnd;
9.                   compute cwnd_incr;
10.          }
11.          else reset(cwnd_incr);
12.  }
13.  cross-layer entity(cwnd_incr);
14.  activate BTP;
15.  compute(SR); send(SR);
16.  END

17.  TRANSPORT::BEGIN (EVENT = recv ACK)
18.  if ((slow-start = TRUE) && (cwnd_incr > 0))
19.  {    cwnd = cwnd + cwnd_incr;
20.       reset(cwnd_incr);        }
21.  END
```

Table 2. Pseudo code for Satellite BoD scheduler

```
1.  MAC::BEGIN (t = allocation time)
2.  While (TS available)
3.  {       allocate(SR, TS);     }
4.  if (total SR > C)
5.  {       buffer unsatisfied SR;}
6.  else {  distribute remaining slots by FCA;   }
7.  construct(BTP); broadcast(BTP);
8.  END
```

The core idea is to have the BoD communicate the latest amount of free capacity allocated to the TCP sender so that the sender knows exactly how much it can open up its $cwnd$ without causing congestion. So, when a BTP is received, each BoD entity will calculate the free slots allocated for its TCP connection(s) based on its record on past requests. The result is translated from slots to TCP segment via (2): -

$$cwnd_i^{incr} = \frac{TS_i^{FCA} \times \left(\theta_{TS} \times F_{period}\right)/8 \times SF_F}{psize_{TCP}} \tag{2}$$

where θ_{TS} is the rate granularity of the slot in kbps , F_{period} is the frame duration in second, SF_F is the number of frames within a super frame and $psize_{TCP}$ is the size of the TCP segment in byte. The computed $cwnd_i^{incr}$ value is truncated to the nearest integer. This information is passed to the cross-layer entity.

When an ACK is received, the TCP sender fetches $cwnd_i^{incr}$ from the cross-layer entity and opens up its $cwnd$ accordingly. The $cwnd_i^{incr}$ will then be reset to avoid

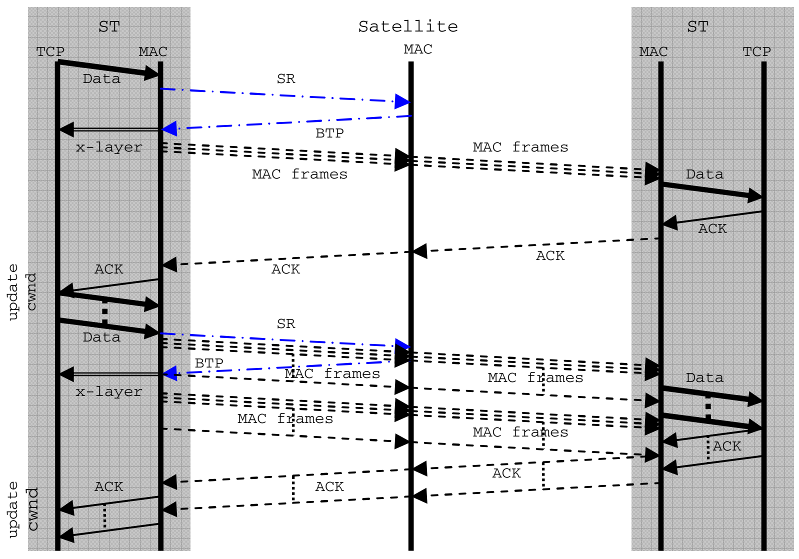

Fig. 4. Timing diagram for interaction between ST and BoD scheduler

multiple increases of $cwnd$. Fig. 4 illustrates the timing for the interaction between the STs and BoD scheduler. Note that, even though the $cwnd_i^{incr}$ has been updated after the first *system response time*, we only allow the TCP sender to open its $cwnd$ when an ACK is received. This ensures that the connection is working properly since reception of ACKs implies that the previous data has been correctly received.

5 Performance Evaluation

5.1 Simulation Setup

We implement our cross-layer algorithm in the *ns-2*[19] simulator with an extension of BoD module. Fig. 2 shows the network topology used where the bottleneck is assumed to be at the satellite segment. Two satellite link capacities are being evaluated: 512kbps and 2048kbps link yielding 32 and 128 16kbps slots per frame respectively. Each frame is of 24ms duration and a super-frame (SF) consists of four frames (i.e. $SF_F = 4$; SF duration = 96ms). We enable the fragmentation function. Each TCP segment will be fragmented into multiple 48 bytes MAC frames; each with a 5-byte header. The receiver MAC reassembles these frames before passing the segment to the upper layer. The link buffers are configured to be big enough to avoid link layer losses. In a BoD network, there are basically three options in how requests are submitted to the scheduler: (1) via pre-assigned slots, (2) by piggybacking on data slots and (3) via contention in a random access paradigm. We use the first option. The FCA option is turned on. Spare capacity is distributed in a fair round-robin manner. We compare our new algorithm with TCP NewReno. The TCP segment size is 500 bytes. The effect of our solution is demonstrated by considering both long-lived and short-lived TCP sources. The receiver is set to acknowledge all segments received.

5.2 Preliminary Inspection of the Mechanism Behavior

Our preliminary evaluation of the scheme involves two simple scenarios where a TCP sender transmits data over an under-utilized network over lossless satellite links set with capacity 512kbps and 2048kbps respectively. The receiver advertised window is set as 64kB. Fig. 5(a) compares the evolution of the $cwnd$ value for the connection with and without our cross-layer mechanism while Fig. 5(b) shows the instantaneous TCP throughput of each case. Clearly, the TCP connection with the cross-layer mechanism manages to open up its $cwnd$ quicker, achieving thus higher throughput in shorter time. Graphically, from Fig. 5(b), the area between the two plots of the same link capacities represents the throughput difference. For instance, the performance gain for the 2048kbps link is 282.5kB for the period between 2s and 8s (the shaded area). However, we also see that the throughput level will coincide thus providing the same performance, implying that the cross-layer mechanism only offers performance gain in the initial phase of the connection (i.e. the slow-start phase). We also see that better bandwidth utilization is only achieved during the slow-start phase.

We further present in Fig. 6 the time sequence of the connections in the two scenarios. In both cases, the cross-layer mechanism gives its TCP sender a “head start” in the packet transmission, obtaining thus a better throughput performance throughout.

We show in section 5.5 that the performance gain increases in lossy satellite links as slow-start events will occur more often. Also, when we compare the two figures, we see that for the 2048kbps satellite link, the performance gain is higher. This is logical since there are more free slots thus enabling bigger increase in the $cwnd$ value. Hence, the performance gain achievable is dependent on the link capacity.

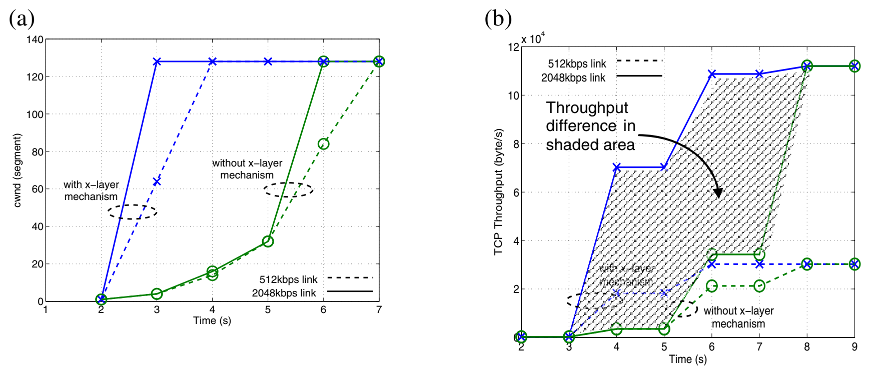

Fig. 5. Comparing (a) the $cwnd$ evolution and (b) throughput achieved for TCP connection with and without cross-layer enhancement over two types of link capacity

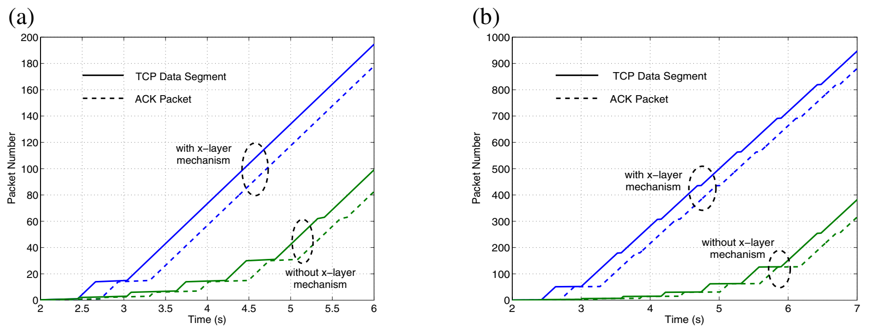

Fig. 6. Slow-start with and without cross-layer for (a) 512kbps and (b) 2048kbps link

5.3 Impact to File Transfer Sessions

We evaluate the performance improvement of our cross-layer mechanism for file transfer. We set up file transfer protocol (ftp) sessions for transferring files of different sizes ranging from 10kB to 10MB across satellite link with capacity 512kbps and 2048kbps. For these simulations, the advertised window is set to a high value (6.4MB). The performance improvements achieved are shown in Fig. 7. We see a general trend of decreasing performance improvement when the file size increases. This is because once the connection exits the slow-start phase, it behaves as the original TCP protocol and yields similar performance for the rest of the transfer time.

Scrutinizing the performance improvement for the case with link capacity 2048kbps, we see a gentle increasing slope for the transfer of very small files. This is due to the fact that since the original transfer times for these files are already short, the completion of the file transfers has been achieved before the congestion avoidance stage was reached. In other words, these cases do not attain the maximum benefit of the cross-layer mechanism. For the case with link capacity 512kbps, we detect an extra slow-start event for file transfer of size 1MB and above. This explains the “spike” in the figure.

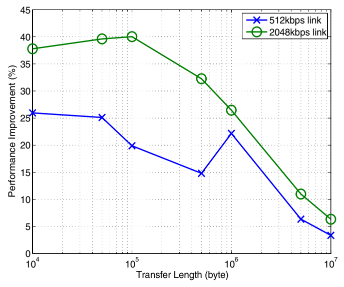

Fig. 7. Performance improvement generally decreases as file size increases

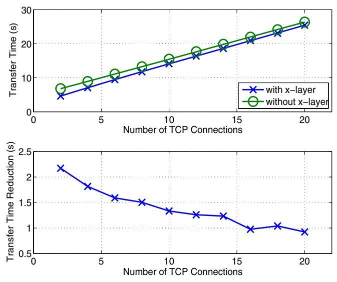

Fig. 8. File transfer time comparison for TCP with and without cross-layer mechanism

5.4 Multiple Competing Connections Scenario

We investigate the aggregate performance of our cross-layer mechanism over error-free satellite channel. Multiple TCP connections are setup to send 100 packets each over a 2048kbps link whereby each connection is started 20ms apart. Intuitively, the effect of the cross-layer mechanism should decrease when there are many competing connections since most TSs will be allocated, resulting in a low number of free slots available. Fig. 8 confirms the deduction. Although the cross-layer mechanism ensures a better performance (Fig. 8 (upper)), its absolute gain decreases with more competing connections (Fig. 8 (lower)). Extrapolating the results, our proposal will not provide performance improvement in a highly congested link since there is no spare capacity. So, it is only beneficial to users in underutilized network condition.

5.5 Lossy Satellite Link Scenario

In the literature, there exist two mainstream approaches in studying a lossy satellite link scenario. The traditional approach is to inject errors to packets causing packet drops. This allows packet level performance examination. The alternative approach is to model the attenuation effect on a satellite link as a decrease of bandwidth [20]. The reasoning behind this approach is that with the advance of forward error correction (FEC) techniques, erroneous packets are recoverable at the expense of redundancy overhead. The worse the link condition, the higher the redundancy overhead used, resulting in less bandwidth devoted to carrying actual information. Since packet-level dynamics are important for our evaluation, we use the first approach.

We configure the 2048kbps satellite link to vary across a range of frame error rate (FER) and compare the time needed to transfer a 10MB file for a connection with and without the cross-layer mechanism in an otherwise unutilized link. We present the results in Fig. 9 which shows that the gain in using the cross-layer mechanism increases exponentially with the FER. This occurrence is directly related to the number of the slow-start event taking place within the duration of the connection.

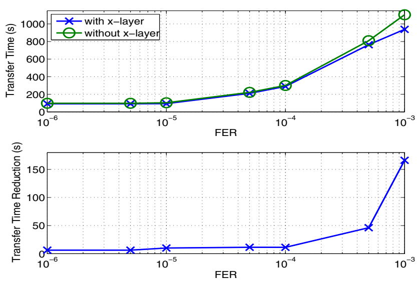

Fig. 9. File transfer time comparison over a lossy satellite link

6 Summary and Conclusions

In this paper, we propose a simplistic cross-layer mechanism between the transport (TCP) and MAC (BoD module) layer to enhance the performance of TCP connections in a GEO satellite network. The proposal speeds up the TCP slow-start by utilizing information acquired from the MAC layer via minimal cross-layer communication. Our proposal provides potentially significant improvements to TCP performance, especially for medium sized file transfers over lossy satellite links. With minimal changes to the original protocol, it is also readily deployable. We also show when the mechanism is of no benefit to both users and satellite operators. From our evaluations, we summarize our findings regarding the mechanism in Table 3 below: -

Table 3. Characterization of the cross-layer mechanism

Operating conditions / Scenario		Gain	Operating conditions / Scenario		Gain
Connection duration	Long	Low	File Transfer (for file transfer)	Big	Low
	Short	High		Small	High
Link Capacity	High	High	Number of connections	High	Low
	Low	Low		Low	High
Link Utilization	High	Low	Link Condition (FER)	High	High
	Low	High		low	Low

There remain some non-trivial practical issues to be considered. First, the mechanism operates under the assumption that the satellite links are the bottleneck link. Although this is usually the case, we have not investigated the penalty incurred (if any) if this assumption is not true. In this sense, its applicability to the Internet is unverified. Nevertheless, our results suggest that it is feasible for deployment in smaller networks that involve a single administrative domain. Second, the more aggressive approach used here will pose questions on fairness as traditional TCP senders are, by design, not able to utilize the free capacity. However, we argue that our proposal does not actually deprive them from the resources. From a different perspective, it is actually an incentive for users to start using our mechanism as early as possible for better performance. In short, we have presented an initial study of the mechanism and showed its potential. Further in-depth investigations, possibly in a testbed, should be conducted to quantify the performance gain.

Acknowledgments. This work is performed within the framework of the SatNEx and ENTHRONE projects, funded by European Commissions (EC) under the Framework Program 6. The financial contribution of the EC towards this project is greatly appreciated.

References

1. Allman, M., Paxson, V., Stevens, W.: TCP congestion control, RFC 2581 (April 1999)
2. Allman, M., et al.: Increasing TCPs initial window, RFC2414 (September 1998)
3. Akyildiz, I., Morabito, G., Palazzo, S.: TCP-Peach: a new congestion avoidance control scheme for satellite IP networks. IEEE/ACM Trans. on Network 9, 307–321 (2001)
4. Casetti, C., et al.: TCP Westwood: end-to-end congestion control for wired/wireless networks. Wireless Networks Journal 8, 467–479 (2002)
5. Caini, C., Firrincieli, R.: TCP Hybla: a TCP enhancement for heterogeneous networks. Int'l J. of Satell. Commun. and Network 22, 547–566 (2004)
6. Henderson, T.R., Katz, R.: Transport protocols for Internet compatible satellite networks. IEEE J. of Selected Areas in Commun. 17(2) (1999)
7. Allman, M., et al.: Enhancing TCP over satellite networks, RFC2488 (January 1999)
8. Caini, C., et al.: Transport layer protocols and architectures for satellite networks. Int'l J. of Satell. Commun. and Network 25, 1–26 (2007)
9. Giambene, G., Kota, S.: Cross-layer protocol optimization for satellite communications networks: A survey. Int'l J. of Satell. Commun. and Network 24, 323–341 (2006)
10. Celandroni, N., Davoli, F., Ferro, E., Gotta, A.: Long-lived TCP Connections via satellite: cross-layer bandwidth allocation, pricing and adaptive control. IEEE/ACM Trans. on Network 14(5), 1019–1030 (2006)
11. Chini, P., Giambene, G., Bartolini, D., Luglio, M., Roseti, C.: Dynamic resource allocation based on a TCP-MAC cross-layer approach for DVB-RCS satellite networks. Int'l J. of Satell. Commun. and Network 24, 367–385 (2006)
12. Peng, F., Wu, L., Leung, V.C.M.: Cross-layer enhancement of TCP split-connections over satellites links. Int'l J. of Satell. Commun. and Network 24, 405–418 (2006)
13. Floyd, S., Jacobson, V.: Early random detection gateways for congestion avoidance. IEEE/ACM Trans. on Network 1(4), 397–413 (1993)
14. Kalama, M., et al.: Cross-layer improvement for TCP Westwood and VoIP over satellite. In: Int'l Workshop on Satellite and Space Commun, pp. 204–208 (2006)
15. ETSI, Satellite Earth Stations and Systems (SES); Broadband Satellite Multimedia (BSM) services and architectures; functional architecture for IP internetworking with BSM networks," TS 102 292, V.1.1.1 (2004-2)

16. ETSI, Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems, EN 301 790, V.1.3.1 (2003-03)
17. Açar, G.: End-to-end resource management in geostationary satellite networks, PhD. Dissertation, University of London (November 2001)
18. Kawadia, V., Kumar, P.R.: A cautionary perspective on cross-layer design. IEEE Wireless Communications, 3–11 (2005)
19. The ns manual [Online]. Available: http://www.isi.edu/nsnam/ns/doc
20. Bolla, R., Davoli, F., Marchese, M.: Adaptive bandwidth allocation methods in the satellite environment. In: Proc. IEEE ICC, pp. 3183–3190 (2001)