TCP and Link Layer Enhancements in DVB-S/DVB-RCS satellite systems

Davide Astuti
Department of Computer Science
University of Helsinki

davide.astuti@cs.helsinki.fi

ABSTRACT

The Transmission Control Protocol (TCP) is efficient on wired networks, but provides poor performance on satellite networks due to the specific characteristics of satellite links. In this paper, we present the experiments results for a combined approach of a selected set of state-of-art TCP enhancements in conjunction with link layer enhancements. Link layer enhancements consist of a link-aware protocol called Satellite Link Aware Communication Protocol (SLACP), which is designed to improve TCP performance over satellite links. We perform experiments in an emulated satellite DVB-S/DVB-RCS environment with a real implementation of SLACP and TCP enhancements on Linux. Results show that both TCP and link-level enhancements significantly improve TCP performance.

Keywords

TCP Performance, Wireless and Satellite Networks, DVB-S/DVB-RCS, FEC, ARQ.

1. INTRODUCTION

The Transmission Control Protocol (TCP) [1] was designed for wired networks and has been highly tuned over the years. Although TCP is efficient on wired networks, it provides poor performance on satellite networks due to the specific characteristics of satellite links and their impact on TCP [2]. As TCP is a widely used transport protocol in Internet, a suitable approach consists of adapting TCP to satellite links rather than design a new transport protocol for satellite networks.

Satellite links have a unique set of characteristics such as high bandwidth-delay product, large round trip time (RTT), high bit error rate (BER) and bandwidth asymmetry [2]. A major problem with TCP has been its inability to distinguish between packet losses due to transmission errors and packet losses due to congestion. TCP supposes that all packet losses are congestion-related and reacts by reducing the sending rate, even though this is necessary only in the latter case.

In this paper, we present the experiments results for a combined approach of a selected set of state-of-art TCP enhancements in conjunction with link layer enhancements. Link layer enhancements consist of a link-aware protocol called Satellite Link Aware Communication Protocol (SLACP), which is designed to improve TCP performance over satellite links [3,4]. We perform experiments in an emulated satellite DVB-S/DVB-RCS environment with a real implementation of SLACP and TCP enhancements on Linux.

SLACP implements error recovery by using FEC and ARQ mechanisms in a novel way that minimizes the additional delay due to retransmission. SLACP retransmissions are based on selective acknowledgments, allowing sender to start retransmission of lost frames with minimal delay. If the link layer is persistent in trying to retransmit lost packets, this would be harmful to the TCP performance, as TCP would start retransmitting the same packets unnecessarily. SLACP sender is not persistent in retransmitting the lost frames, instead it retransmits a lost frame only a limited number of times. At the same time it sends FEC-encoded redundant data with the retransmitted frames to increase the probability of successful delivery of retransmitted frames within one RTT. This together with proper prioritization retransmissions allows fast recovery avoiding interactions between the link-layer and transport-layer timers.

SLACP incorporates other important and useful features. One important feature is flow control between the IP layer and the link (MAC) layer together with limiting the amount of link buffering. In many wireless systems, such as DVB-S/DVB-RCS satellite systems, IP packets flow directly to the MAC buffer without flow control between the layers and excess packets are dropped if the MAC buffer becomes full. The IP queue will always be empty; thus, proper router queue sizes to control total amount of buffering cannot be used and using IP active queue management mechanisms is not effective. In our approach, new arriving packets are forwarded to the MAC buffer only if space is available; otherwise, packets are kept in the IP buffer. In addition, any unnecessary link-level buffering is avoided to minimize the overall delay.

SLACP has been implemented in Linux and extensively tested using a satellite emulator, called Network Engineering Platform (NEP) from Alcatel. The NEP platform is representative of a DVB-S/DVB-RCS satellite system focusing on the networking functions. The satellite platform is used to emulate several levels of error rate and a Demand Assignment Multiple Access (DAMA) bandwidth on demand allocation scheme on the satellite return link. Tests using three kinds of DAMA bandwidth allocation scheme - CRA, RBDC and VBDC - have been carried out.

We show that employing the selected set of TCP enhancements decreases the transfer time by 5-68% depending on the link error rate. Employing SLACP on a lossy link is extremely beneficial to TCP, decreasing the transfer time by 5-91%. Further, combining the use of TCP enhancements with SLACP gives the best performance.

The rest of this paper is organized as follows. Section 2 provides an overview of SLACP design principles. Section 3 describes the selected TCP improvements and Section 4 describes experiments and performance results. Finally, conclusions are provided in section 5.

2. SLACP OVERVIEW

Satellite-Link Aware Communication Protocol (SLACP) [3,4] is specifically designed for use over satellite links to enhance the transport performance. SLACP logically operates at the link layer and includes various useful features for mapping IP layer services to the underlying satellite link services.

The principal design elements of SLACP are the mechanisms used for the provision for QoS, the flow control between MAC layer and IP layer, and the error recovery. The provision of QoS is a major issue when the underlying satellite link is intended to be used for real-time and delay sensitive traffic such as voice and multimedia. SLACP implements several logical channels with different QoS parameters and error control strategies. All packets belonging to an IP flow are directed to a SLACP channel with appropriate QoS parameters for that IP QoS class. Frames transmitted over each logical channel are delivered independently of the frames sent over the other channels. A single high priority channel, called control channel, is reserved for retransmission of frames, SLACP acknowledgments and control frames.

In DVB-RCS satellite systems IP packets flow directly into the MAC buffer since the bandwidth allocation is made based on the amount of data in the MAC buffer. The drawback of this approach is that the IP queue is kept empty and the buffering is made only at the MAC layer. Therefore, proper router queue sizes to control total amount of buffering cannot be used and using IP active queue management mechanisms is not effective. When SLACP is used, new arriving packets are forwarded to the MAC buffer only if space is available; otherwise, packets are kept in the IP buffer. Doing the buffering in the IP layer does not avoid packet drops in case of congestion (IP buffer may also overflow) but gives advantages because active queue management mechanisms, i.e. RED, can be applied. In addition, any unnecessary link-level buffering is avoided to minimize the overall delay.

The SLACP supports both FEC and ARQ mechanisms for recovery from frame losses due to bit errors in the satellite channel. There are four possible options for error recovery - no error control scheme (no ARQ, no FEC), FEC only, ARQ only and FEC-ARQ (ARQ and FEC combined). The error control scheme is configured separately for each channel. In SLACP, FEC-ARQ is the default error recovery scheme.

Retransmissions are done through a FEC-protected channel with high priority. The frames for retransmission from all ARQ-enabled channels are collected and sent through the same high priority channel. This will reduce the delay on retransmitted data as higher priority frames would be scheduled first for transmission. By default only a single retransmission attempt requiring one round trip time is used. FEC is used to add more redundancy to the retransmitted data.

In SLACP, Reed-Solomon code is used for FEC encoding of frames in a novel way. FEC encoding frames are organized as FEC blocks. A FEC block consists of actual frames and redundancy frames as shown in Figure 1. An additional FEC header is added to each frame. Reed-Solomon code is specified through the parameters [n, k, n-k+1] where n is the length of the codeword, k the dimension of the code. This scheme, as shown in Figure 1 is used to protect a set of k SLACP actual frames (IP packets plus SLACP headers). From these frames, we construct n-k frames of redundancy. We consider that each SLACP frame has a maximum length L. If an SLACP frame has a length less than L, for constructing the redundancy frames virtual padding (zero padding) bytes are used to make its length to L. These padding bytes are used to construct the redundant frames. The sender transmits the frames without these padding bytes.

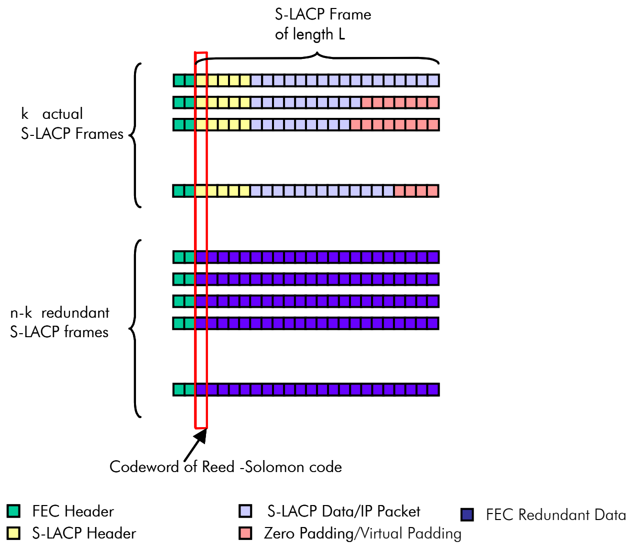


Figure 1. Scheme of SLACP frame encoding using FEC with k actual frames.

The construction of the n-k redundant frames from the k frames is done by considering the i^{th} byte of each frame and by constructing the corresponding Reed-Solomon code for i^{th} byte in redundant frames (Figure 1). The codewords are framed vertically to construct the redundant bytes but transmission over the link is performed horizontally one frame at a time, that is, all the bytes of a frame are sent before the next frame.

In a FEC encoded channel, the sender can transmit the actual frames immediately. The redundant frames are constructed, once k original frames have been sent or a timer expires. In the latter case the code block consists of less than k frames and zero padding frames are used to add the number of actual frames to be equal to k and then compute the redundant data for each codeword. The SLACP sender will not send these zero padding frames.

Further optimization is used to reduce the number of bytes in redundant frames when a FEC block consists of frames with different length. For example, if a FEC block has only a few longer frames, the longest having length L, and the rest of the actual frames have length < = S, it is enough to send only a limited number of redundant frames of length L while the rest of the redundant frames have length of S. In this study we have implemented such a FEC bandwidth optimization with three frame length categories.

3. TCP IMPROVEMENTS

In this section we describe several techniques that we have selected for enhancing the performance of TCP.

Window Scaling [5] extension allows TCP to support larger windows. It expands the definition of the TCP window to 32 bits and then uses a scale factor to carry this 32-bit value in the 16-bit Window field of the TCP header.

Increasing the initial window from 1 up to 4 segments [6] increases the number of segments sent during the first RTT of the connection, allowing more rapid opening of the congestion window. This is extremely useful for high latency environments.

Delayed ACK After Slow Start (DAASS) [7] consists of sending acknowledgment for every segment during slow start so that the congestion window grows up faster.

Limited transmit [8] allows the sender to transmit a segment for each of the first two duplicate acknowledgments. This is useful for TCP connections with small congestion windows or when a large number of segments are lost in a single transmission window as otherwise it may happen that not enough duplicate acknowledgment arrive at the sender, the fast retransmit algorithm will not be triggered and TCP must wait for a costly retransmission timeout.

SACK [9] based mechanisms allow TCP to recover more efficiently from multiple segment losses in a window of data.

Forward RTO Recovery (F-RTO) [10] algorithm effectively helps detecting spurious TCP RTOs and avoiding unnecessary retransmissions and thereby improves TCP performance in the presence of delay spikes. This is very useful as delay spikes may occur as ARQ and dynamic link bandwidth allocation mechanisms are used.

TCP Control Block Interdependence (CBI) [11] aims to share part of the TCB (TCP Control Block) to improve TCP performance. TCB is a data structure associated with every TCP connection which contains the information about the connection state such as RTT estimate, congestion window size, slow-start threshold (ssthresh). In particular, sharing the ssthresh value from a previous connection allows a TCP sender to use this value when initializing the ssthresh for a new connection and thereby avoid router buffer overshoot in the end of initial slow-start. How-ever, this can create problems of using too slow initial ssthresh value when ssthresh of a previous connection is set by the occurrence of a packet loss due to link error.

4. PERFORMANCE EXPERIMENTS

We ran experiments to evaluate the performance of the TCP enhancements combined with SLACP. We used TCP on Linux end hosts and a satellite emulator. We compare the performance of a regular and enhanced TCP version. We also compare the TCP performance with and without SLACP.

The test environment consists of a satellite end-host that communicates with another a host via a router connected to a satellite link. Figure 1 shows an overview of the network topology. The satellite end-host (H1) accesses the emulated satellite network through the Satellite Terminal (ST). The other end host (H2) is connected to the satellite network through the Broadband Access Server (BAS). The ST and BAS act as routers. The direction from H1 to H2 is referred to as Return Link; the other direction is referred to as Forward Link.

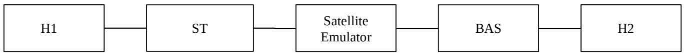


Figure 2. Network Topology.

We used a satellite emulator platform which is representative of a DVB-S/DVB-RCS satellite system. The DAMA bandwidth allocation scheme, which is applied on the Return Link, is accurately emulated. The DVB-RCS standard defines four classes of traffic assignment.

Constant Rate Assignment (CRA) allocated bandwidth is guaranteed and requires no dynamic signaling. The traffic is not subjected to any scheduling/queuing delay.

Rate Based Dynamic Capacity (RBDC) is allocated based on two components: a pre-fixed ceiling rate and the instantaneous rate request sent by the ST. The ST can request bandwidth up to the pre-fixed ceiling value. A bandwidth request remains effective until it is updated by ST or until it is timed out. During the time a bandwidth request remains effective, the ST traffic is not subjected to any additional bandwidth allocation delay. In contrast to CRA, RBDC strategy allows for statistical multiplexing among many terminals, resulting in a more efficient use of satellite bandwidth.

Volume Based Dynamic Capacity (VBDC) is assigned in response to a request by the ST. The bandwidth request is based on the amount of data waiting in the MAC buffer. A scheduler accumulates all requests against each ST it serves. As bandwidth is shared by a number of terminals there is no guarantee on capacity availability for a specific bandwidth request. The scheduler assigns capacity from queue of requests, within the constraints of the remaining capacity after CRA and RBDC assignments. This strategy provides an efficient use of network resources but traffic may experience a significant delay due to such a bandwidth allocation mechanism.

Free Capacity (FCA) is the remaining bandwidth left after all other bandwidth re-quests have been served. It is assigned among all STs in a round-robin fashion.

The emulated bandwidth on the satellite Forward link is 512 Kbit/s and on the Return link 128 Kbit/s. In our experiments we used the CRA, RBDC and VBDC schemes (see Table 1). The MTU size is 1450 bytes and the one-way propagation delay is set to 250ms. The bandwidth between H1 and ST as well as H2 and BAS is 100 Mbit/s.

Table 1. Network Configuration

Scheme	Forward	Return			
	Peak Bandwidth	guaranteed access bandwidth		non guaranteed access bandwidth	
VBDC	512Kb/s	n/a	n/a	VBDC	128Kb/s
RBDC	512Kb/s	RBDC	128Kb/s	n/a	n/a
CRA	512Kb/s	CRA	128Kb/s	n/a	n/a
CRA&VBDC	512Kb/s	CRA	8Kb/s	VBDC	120Kb/s

The satellite MAC buffer size has been tuned experimentally to allow the maximum utilization of the link bandwidth in case of VBDC. As discussed earlier, regular IP-to-satellite interface typically does not provide flow control between the layers and all buffering effectively occurs at the MAC layer. However, a certain amount of buffer space is needed to allow the TCP congestion window to grow up and correctly estimate the available link bandwidth. When SLACP was not used, the MAC buffer size of 600 ATM cells (28800 bytes) was enough. This is roughly the link bandwidth-delay product taking into account an RTT of 2 seconds (typical RTT values with VBDC are between 1.3-2 sec). When SLACP was used, we could reduce the MAC buffer to 300 ATM cells (14400 bytes). The IP queue size was 40 Kbytes.

In order to model the satellite errors, we consider that the link experiences bursty errors. This is modeled with a two-state Markov model with an error-free good state and a bad state where error burst corrupts all packets. We use uniform distribution for both burst inter-arrival time (good state duration) and burst length. Five error models are used in this study. The error model parameter values are shown in Table 2.

Table 2. Satellite Error Models

Error Model	Time between Bursts		Burst Length	
	Min	Max	Min	Max
None	Link without any error			
Low	10	30	60ms	60ms
Medium	5	15	100ms	300ms
High	2	10	100ms	300ms
Huge	0	7	100ms	100ms

We modified the TCP implementation in Linux so that it implements three TCP variants: Reference TCP, Satellite TCP and Enhanced TCP. The Reference TCP implements regular TCP behavior with SACK disabled, delayed ack threshold set to 200ms, and the initial window set to one segment.

The Satellite TCP consists of Reference TCP extended with standard TCP mechanisms recommended for use over satellite links [2]: Conservative SACK algorithm [12], initial window of two segments and Window Scaling option.

The Enhanced TCP enables the TCP enhancements discussed in section 3, that is, initial window of 4 segments, SACK, Limited Transmit, DAASS, F-RTO and Window Scaling. The DAASS option is implemented by enabling the Quick ACKs in Linux. In addition, a subset of tests was repeated with the TCP Control Block Interdependence (CBI) turned on. A single unidirectional TCP bulk-data connection is used to transfer 1 MB from H1 to H2. We repeated the same set of tests transferring the same amount of data with four parallel TCP connections (4x250 Kbytes) to verify the results in presence of competing TCP connections. Each test case is replicated 32 times. In this paper, we report a subset of the test results of 1MB bulk-data transfer only for brevity reasons. All tests results are reported in [3,4].

4.1 SLACP Protocol Evaluation

This section analyzes the SLACP protocol behavior during two typical packet recovery phases: ARQ-based and FEC-ARQ based. In the graphs, a series of plus signs connected with an orange line indicates the transmitted frames. A similar series connected with blue line represents the frames received, and a light blue diamond represents the delivery of a frame to the higher layer (IP). The green line indicates received acknowledgements, a red plus marked with 'Rs' above it marks a sent retransmission, and a red plus marked with 'Rr' shows a received retransmission. 'Fa' marks an acknowledgement recovered with FEC and 'Fd' marks a FEC recovered retransmission of a data frame.

Figure 3 shows a typical recovery using ACK information. Data frames with sequence numbers till 6713 are sent, correctly received and delivered to IP layer. Frames from 6714 to 6716 have been lost in the direct transmission. The receiver will send an acknowledgement containing selective acks information immediately after it notices that some frames have been lost. When an acknowledgement informing the losses arrives at the sender (the green line), the lost frames are retransmitted immediately (marked with a red Rs). SLACP does not deliver the frames out of order in order to being TCP friendly, but stores the frames for later delivery in case the frames arrive or SLACP gives up retransmitting. As the figure shows, frames from 6717 to 6723 are delivered to IP only after the retransmissions have been received correctly, so that packet reordering is avoided.

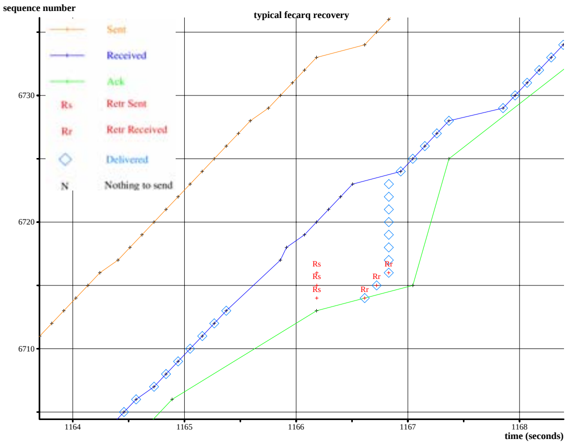


Figure 3. Typical ARQ-based recovery

SLACP uses a FEC (forward error correction) scheme to protect its control frames and retransmissions. Figure 4 shows a case where data frames from 11159 to 11165 have been lost during the first transmission and the retransmissions of frames from 11161 to 11164 have been lost again. Since the retransmissions are protected with FEC redundancy (sent after retransmissions when the current FEC block is full or a timer expires), SLACP is able to recover the lost frames when enough redundant data has arrived. Data frames recovered because of FEC are marked as Fd in the graph. FEC recovery also preserves order, so the correctly received retransmissions (frame from 11165) is not delivered to the IP layer before all data frames in the current FEC block have been recovered.

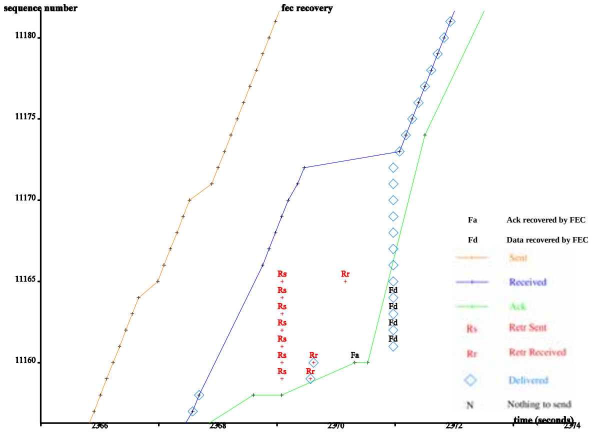
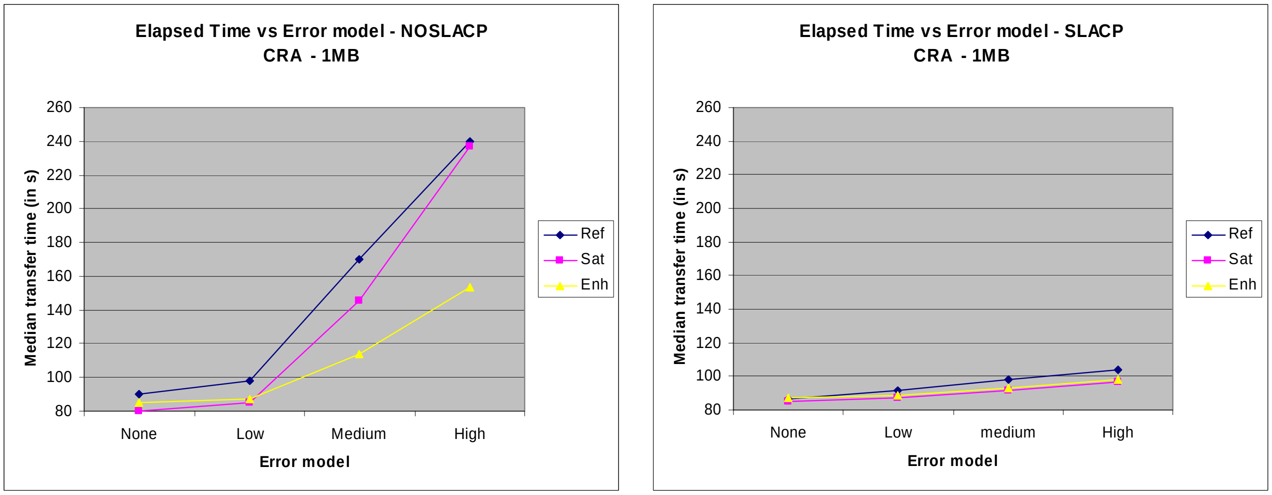


Figure 4. Typical FEC-based recovery



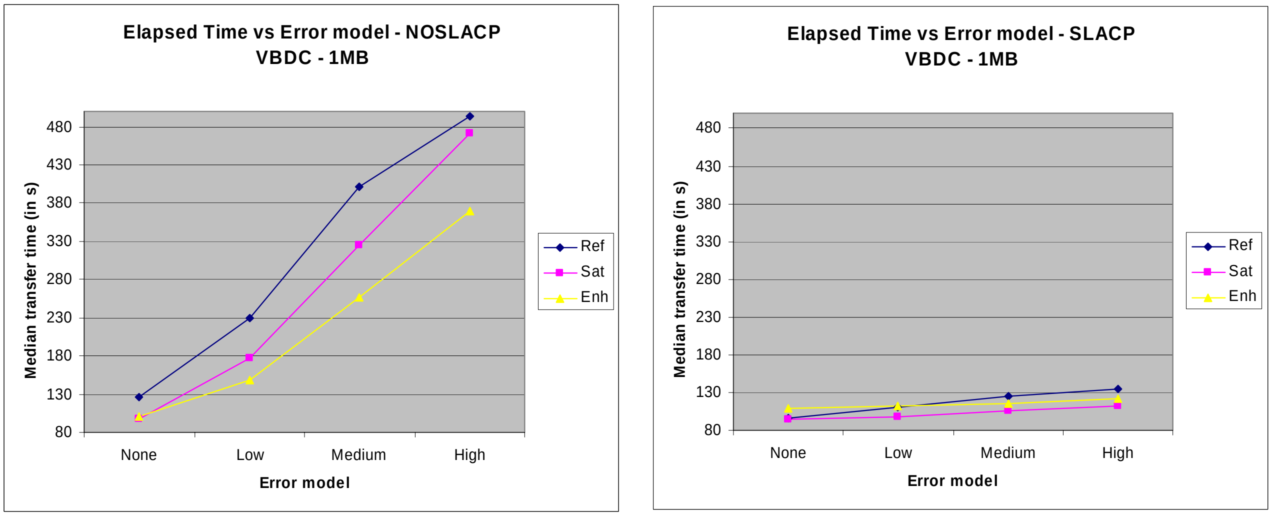
(a) (b)

Figure 5. Performance of Reference TCP, Satellite TCP and Enhanced TCP, with and without SLACP, CRA.

4.2 Results

Figure 5 and Figure 6 shows the performance of the three TCP versions without SLACP (part a) and with SLACP (part b). The graphs show the median transfer time of the 32 replications for four error models when CRA (Figure 5) and VBDC (Figure 6) bandwidth allocation schemes are used. In general, when SLACP was not employed, the Enhanced TCP and Satellite TCP perform better than Reference TCP for all error models and for all DAMA allocation schemes. The Enhanced TCP yields clearly increasing performance gain when the error rate becomes higher. Comparing experiments with CRA and VBDC shows that as the DAMA allocation delay increases, the TCP performance decreases for all three TCP versions and the improvement due to TCP enhancements only is not that significant in case of VBDC (figure 6a high error) compared to the case of CRA (figure 5a high error). TCP performance is strongly affected by such a delay as high RTT slows down TCP loss recovery and the congestion window increase. However, Enhanced TCP suffers less from the increasing delay. Even on error-free link Enhanced TCP is slightly more efficient than Reference TCP as it is able to inflate the congestion window faster. The larger initial window and the DAASS mechanism present in Enhanced TCP have much dramatic effect when the transfer size is smaller, like in most Web transfers. With VBDC and Enhanced TCP on an error-free link, the first 10 Kbytes of data are transferred in 4.7 sec compared to 10 sec of Reference TCP; the first 50 Kbytes of data are transferred in 8.2 sec and 13.7 sec, respectively.

The use of SLACP improves TCP performance considerably, especially at high error rates (compare see Figure 5a with 5b and 6a with 6b). This is true for all three TCP versions. When the error rate increases, it becomes obvious that Enhanced TCP alone is not able to provide adequate performance; the Reference TCP with SLACP is able to provide significantly better performance compared to Enhanced TCP without SLACP. Even at low error rates (error models *none* and *low*), SLACP is beneficial. This is because of flow control between the IP and the MAC layer. Without SLACP, IP packets flow directly to the MAC buffer. This prevents the utilization of IP buffering. With SLACP, the



(a) (b)

Figure 6. Performance of Reference TCP, Satellite TCP and Enhanced TCP, with and without SLACP, VBDC.

new arriving packets are kept in the IP buffer if no space is available in the MAC buffer. This does not avoid packet drops in case of congestion (IP buffer may also overflow) but allows the use of the available IP buffer space.

Combining Enhanced TCP with SLACP gives the best performance compared to the other cases, except in the case of error-free link. In the presence of errors, the combined approach (Enhanced TCP with SLACP) performs dramatically better than the baseline approach (Reference TCP without SLACP).

In some cases, Reference TCP and Satellite TCP perform slightly better than Enhanced TCP as the latter is more aggressive during the initial slow-start phase and experiences a larger number of congestion-related losses. During the initial slow start the TCP sender increases its congestion window exponentially to probe the available link bandwidth. This results in overshooting the available link bandwidth and loss of multiple packets, requiring a long lasting recovery. This phenomenon is known as slow-start overshoot. This problem is more serious in the case of Enhanced TCP as it uses a larger initial window and the DAASS mechanism. Figure 7 shows an example of slow-start overshoot. The graph shows the packet trace of a TCP connection from the sender point of view. The black line indicates the sequence number of the packets sent; the green line shows the ACKs; the cyan color shows the SACK information carried by the ACKs; the red 'R' indicates retransmissions. The overshoot happens roughly at t = 18 sec, when the sender has transmitted approximately 250 Kbytes. A large number of packets are lost and the recovery phase takes more than 30 seconds.

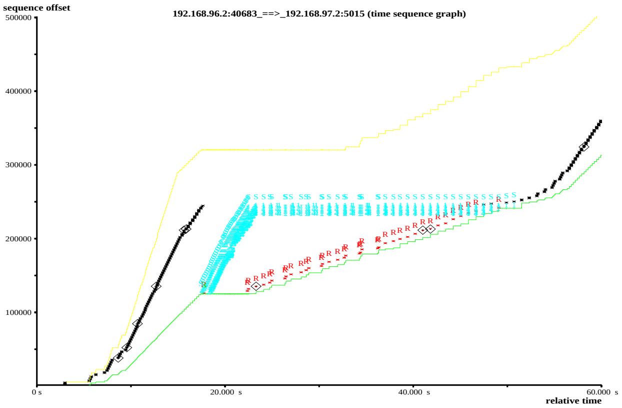


Figure 7. Example of slow-start overshoot.

It turned out that the slow-start overshoot remained the only major problem with the combined approach of using SLACP and Enhanced TCP. In order to attack the slow-start overshoot problem, we used two modified versions of Enhanced TCP. The first version uses the TCP Control Block Interdependence (CBI), which allows the TCP sender to reuse the slow-start threshold of the previous connection. The use of CBI can be better justified when there are little or no uncorrected link errors (which is the case when SLACP is used). The second TCP version does not include the option DAASS, so that it is less aggressive during the initial slow-start phase. Figure 8 shows the median transfer elapsed time of four TCP versions when SLACP is used. Enhanced TCP without DAASS performs better than Enhanced TCP. In general, Enhanced TCP with CBI (and DAASS) performs the best because CBI prevents the overshoot problem and DAASS makes the slow start phase quicker. On the other hand, TCP with CBI does not perform so well in presence of many errors related losses (see Figure 8 huge error model) as CBI may lead to use a too small initial ssthresh value when the ssthresh of a previous connection is set by the occurrence of a packet loss due to a link error. Anyway, this situation is not likely to happen when SLACP is used as it recovers most of packet errors.

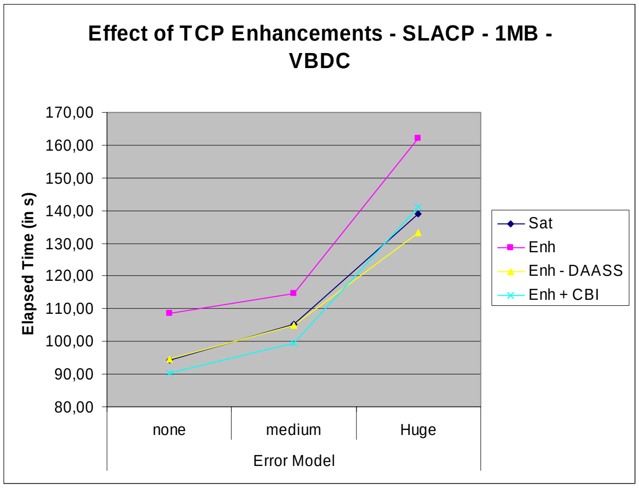


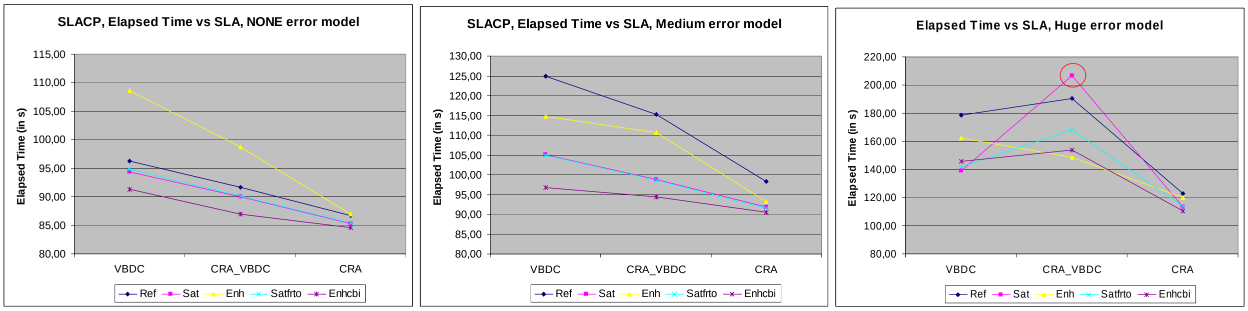
Figure 8. Impact of TCP enhancements on slow-start overshoot phenomenon.

4.3 Effect of DAMA allocation delay on TCP

In our configuration, the VBDC bandwidth allocation scheme typically requires 1.3-2 seconds for allocating the requested amount of network resources. In some cases, such a delay may be over 3 seconds, which would lead to the retransmission of the initial SYN segment of the TCP connection as the initial RTO expires after 3 seconds (standard initial value of RTO). The SYN is retransmitted after 3 seconds because the TCP sender assumes that the SYN is lost even though the SYN was delayed because of the allocation delay. This makes the data transfer starting after roughly 4 seconds. Having even a small amount of CRA in this case prevents the unnecessary retransmission of initial SYN segments and, more importantly, would significantly improve the TCP performance.

Figure 9 shows the effect of using a small amount of CRA (8 Kb/s) together with VBDC (120 Kb/s) compared to pure VBDC (128 Kb/s) and pure CRA (128 Kb/s) for five versions of TCP: Reference TCP, Satellite TCP, Enhanced TCP, Enhanced TCP with CBI and Satellite TCP with the F-RTO algorithm enabled. Figure 9a,9b and 9c shows results for the three error models none, medium and huge.

Figure 9a and 9b shows that the combined use of CRA and VBDC is beneficial to TCP: the median transfer time in this case is in the middle between the ones of pure CRA and pure VBDC. TCP performance is very sensitive to RTT variations and performs better if the RTT maintains stable. Even though using a small amount of CRA with VBDC increases the RTT variability (the RTT as seen by TCP is lower during the beginning of the connections and higher later on), TCP takes advantage of the CRA bandwidth availability. On the other hand, Figure 9c shows that TCP versions and especially Satellite TCP performance is decreased in case CRA and VBDC are used together. As outlined



(a) (b) (c)

Figure 9. Effect of using a small amount of CRA together with VBDC in case of none, medium and huge error model for various TCP versions.

before, TCP is sensitive to RTT variations. Differently from the case where none and medium error models are used (Figure 9a and 9b), in this case RTT variations occur for two reasons: the presence of a little amount of CRA and the huge errors present in the link. A huge number of packet errors occur and packet are retransmitted and recovered by SLACP causing an additional delay. These two factors together make spurious RTO expirations more likely to occur and, hence, unnecessary TCP retransmissions. Simply using Satellite TCP with the F-RTO algorithm enabled (Figure 9c, Satfrto) copes with this problem.

5. CONCLUSIONS

This paper presented the experimental results for the use of a TCP/IP friendly link-level recovery mechanism called SLACP in conjunction with a number of state-of-art TCP enhancements to enhance the performance of TCP on DVB-S/DVB-RCS satellite systems. While SLACP allows efficient TCP operation over lossy wireless links it minimizes the additional delay due to the ARQ by adding redundancy in the retransmitted frames in a novel way. SLACP also incorporates flow control between IP and MAC layer which limits the amount of the link buffering and allows the use of queue management at IP layer.

We ran a set of experiments to evaluate performance of SLACP using real TCP implementations on Linux end hosts and a satellite emulator platform representative of a DVB-S/DVB-RCS satellite system. We compared the performance of baseline version and various version of TCP on an emulated satellite link, with and without SLACP. Using SLACP is extremely useful, in particular on a lossy satellite link as TCP performance is much better when SLACP is used due to the lower residual link-error rate and the flow control between IP and MAC layers.

We found out that both SLACP and TCP enhancements are beneficial on their own and that the combined approach with TCP enhancements and SLACP yields the best performance. Enhanced TCP performance suffers from the slow-start overshoot phenomenon when the residual link-error rate is zero. Introducing TCP Control Block Interdependence mechanism for sharing slow start threshold is helpful to attack the problem.

The large VBDC bandwidth allocation delay impacts over TCP performance. Using even a small amount of CRA capacity together with VBDC significantly improves the performance, although the increased RTT variation may cause spurious TCP retransmission in case of high packet error rates. The F-RTO algorithm is helpful in this case.

6. ACKNOWLEDGEMENTS

This work has been carried out in the Department of Computer Science at University of Helsinki as a part of the TRANSAT project of European Space Agency. We thank all the project members, which supported in carrying out this work.

7. REFERENCES

[1] J.B.Postel, "Transmission Control Protocol," RFC 793, September 1981.

[2] M. Allman, et all “Enhancing TCP Over Satellite Channels using Standard Mechanisms”, RFC 2488, January 1999.

[3] M. Kojo, D. Astuti, L. Daniel, A. Nyrhinen and K. Raatikainen, “Enhancing TCP Performance Over Satellite Networks - A Link-Aware Approach”, to appear.

[4] M. Kojo, D. Astuti, L. Daniel, A. Nyrhinen and K. Raatikainen, “Improving TCP Performance over Wireless WANs using TCP/IP-Friendly Link Layer”, to appear.

[5] V. Jacobson, R. Braden, D. Borman, “TCP Extensions for High Performance”, RFC 1323, May 1992.

[6] M. Allman, S. Floyd, C. Partridge, “Increasing TCP's Initial Window”, RFC 3390, October 2002

[7] M. Allman et al, “Ongoing TCP Research Related to Satellites,” RFC 2760, February 2000.

[8] M. Allman et all, “Enhancing TCP's Loss Recovery Using Limited Transmit”, RFC 3042, January 2001.

[9] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow. “TCP Selective Acknowledgement Options”. RFC 2018, October 1996.

[10] P. Sarolahti, M. Kojo, and K. Raatikainen, “F-RTO: An Enhanced Recovery Algorithm for TCP Retransmission Timeouts”, ACM SIGCOMM Computer Comm. Review, April 2003.

[11] J. Touch, “TCP Control Block Interdependence”, RFC 2140, April 1997.

[12] E. Blanton, M. Allman, K. Fall and L. Wang, “A Conservative Selective Acknowledgment (SACK)-based Loss Recovery Algorithm for TCP”, RFC 3517, April 2003.