Performance Evaluation 66 (2009) 287–300

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

Analysis of TCP live experiments on a real GEO satellite testbed☆

C. Caini [a], R. Firrincieli [a,*], D. Lacamera [a], T. de Cola [b], M. Marchese [c], C. Marcondes [d], M.Y. Sanadidi [d], M. Gerla [d]

[a] DEIS, University of Bologna, Italy
[b] DLR, German Aerospace Center, Germany
[c] CNIT, University of Genoa, Italy
[d] UCLA, Los Angeles, USA

ARTICLE INFO

Article history:
Received 31 January 2008
Received in revised form 23 July 2008
Accepted 31 October 2008
Available online 20 November 2008

Keywords:
Transport protocols
Satellite communications
Testbed
Performance Enhancing Proxy (PEP)
PEPsal
TCP Hybla
TCP-splitting

ABSTRACT

The paper describes a measurement campaign carried out by the University of Bologna (UoB), the National Inter-University Consortium for Telecommunications (CNIT) and the University of California Los Angeles (UCLA). The aim of the experiments was the performance assessment of a wide range of TCP enhancements on network environments that include a real GEO satellite link. To this end, UoB and CNIT integrated their network tools and set up a testbed composed of a cluster of UoB Linux PCs connected to the CNIT GEO Skyplex satellite platform. Tests were conducted considering both end-to-end TCP enhancements and a TCP-splitting Performance Enhancing Proxy (PEP) developed by UoB. The analysis was not limited to isolated satellite links, but embraced more complex heterogeneous networks, where satellite connections have to compete with wired cross traffic for the network resources. The analysis of the large set of experimental data presented in the paper confirms the challenges posed by GEO satellite channels, which, in this case, were worsened also by the presence of Bandwidth on Demand technique, which impacts TCP performance. Among the end-to-end TCP variants, best results are generally shown by TCP Hybla, especially when heterogeneous environments are considered. On the other hand, the splitting PEP solution offers the best performance with respect to all TCP enhancements, due to its ability to isolate satellite channel impairments. The possible drawback of this approach is in that it violates the end-to-end semantics and is incompatible with IPsec protocol implementation.

1. Introduction

The paper describes a measurement campaign recently carried out by the University of Bologna (UoB), the National Inter-University Consortium for Telecommunications (CNIT) and the University of California Los Angeles (UCLA) on the CNIT GEO Skyplex satellite platform [1]. As it is widely recognized [2], satellite channels are really challenging for transport protocol performance, because of their long Round Trip Time (RTT) (more than 600 ms for bidirectional GEO link) and the possible presence of channel errors (depending on satellite platform characteristics and propagation conditions). Both these aspects severely impair transport layer performance when an application requires reliable service. To cope with these issues, several solutions have been proposed. They basically can be classified into two main categories [2,3]: *TCP enhancements*,

☆ This paper was presented in part at IEEE ISCC 2007 conference, Aveiro, Portugal, July 2007.

* Corresponding author.
E-mail addresses: ccaini@arces.unibo.it (C. Caini), rfirrincieli@arces.unibo.it (R. Firrincieli), root@danielinux.net (D. Lacamera), tomaso.decola@dlr.de (T. de Cola), mario.marchese@cnit.it (M. Marchese), cesar@cs.ucla.edu (C. Marcondes), medy@cs.ucla.edu (M.Y. Sanadidi), gerla@cs.ucla.edu (M. Gerla).

0166-5316/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2008.10.012

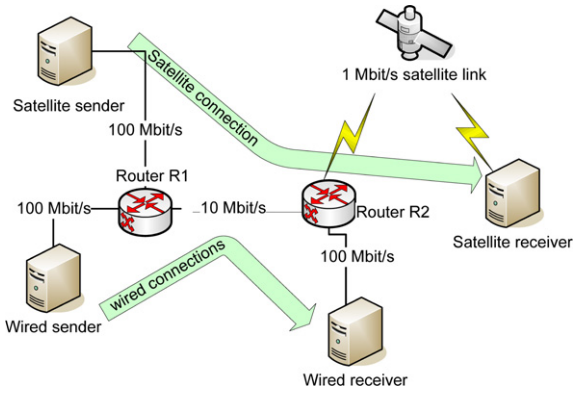


Fig. 1. Testbed logical layout.

which involve only the end hosts, and *Performance Enhancing Proxies (PEPs)*, which introduce intermediate agents on the path, such as those based on TCP-splitting or TCP spoofing [4] concepts. Protocol enhancements generally offer limited performance gain and require sender side modifications on content provider servers. PEPs are the preferred choice for commercial solutions due to their high performance and deployment simplicity (by contrast to the previous case, they are transparent to end hosts). However, they present some major drawbacks related to the violation of end-to-end semantics and the use of IPsec [4].

In the paper, a live measurement campaign on a real satellite platform is presented with the aim of highlighting the challenges posed by real GEO satellite channels, as well as the impact of a dynamic bandwidth assignment technique on TCP performance. Several TCP variants, such as SACK [5], BIC [6], Vegas [7], Hybla [8], Westwood+ [9] and Westwood [10], are considered. The rationale for this choice is that SACK is possibly the most widely adopted variant; BIC is characterized by an innovative congestion control based on binary search; Vegas is an interesting example of delay-based (instead of loss-based) congestion avoidance algorithm; Hybla is specifically designed for satellite channels; finally, Westwood and Westwood+ are intended for wireless channels. Among PEP solutions, the focus is on techniques based on the TCP-splitting concept, and in particular on PEPsal [11], developed by the authors and, to the best of their knowledge, the sole integrated TCP-splitting solution available for Linux OS under GNU GPL (General Public License). Two different scenarios are taken into account: pure satellite networks and heterogeneous networks. In the former, only satellite connections (i.e. connections that include at least one satellite leg) are considered. In the latter, both satellite and entirely wired connections are present. This second scenario, which is quite typical of Internet communications, is the more demanding, because of the severe RTT-unfairness against long delay satellite connections. Measurements, which resulted in more than 5 GB of row data collected in over two weeks of experiments (5 people, 213 tests), were carried out by means of the integrated UoB-CNIT testbed [12,13], whose logical layout is sketched in Fig. 1. All TCP enhancements, as well as PEPsal, were used with their default parameter setting. In particular, PEPsal made use of SACK on the first leg (from the satellite sender to the router R2), and Hybla on the satellite leg (from R2 to the satellite receiver).

The structure of the paper is as follows. In the next section a brief overview of the different solutions evaluated in our test is given. Then, in Section 3 the integrated testbed used for live experiments is described. Next, in Section 4, evaluation tools and measurement objectives are introduced. Sections 5 and 6 present the experimental results, considering pure satellite networks and heterogeneous networks respectively. Conclusions are drawn in Section 7.

2. TCP variant overview

In this section, a brief overview of the different solutions considered in the experiments is given for the convenience of the reader. It is intended as a basic introduction to the solutions examined; the interested reader is referred to the cited literature for further information.

For a better understanding, it is worth summarizing the causes of main problems faced by standard TCP variants (e.g. Reno, NewReno) on GEO satellite channels [2]. First, the standard TCP congestion control suffers from the long RTTs typical of GEO (around 600 ms). This is due to the fact that if the additive constant increase policy is left unaltered, the actual congestion window (cwnd) growth rate is determined by the connection RTT (in the congestion avoidance phase the cwnd is incremented by roughly one segment per RTT): the longer the RTT, the slower the cwnd (re-)opening. By bearing in mind that the transmission rate (in segments) is given by the cwnd over RTT ratio, it is not surprising that long RTT connections, such as satellite, are severely penalized in a variety of circumstances (e.g. loss recovery, congestion in heterogeneous networks, short file transfers). Second, the possible presence of random losses due to the wireless channel may cause significant performance degradation, as they trigger "spurious" (i.e. not justified by a real network congestion state) halving of the cwnd, and consequently of the transmission rate. Even in this case, the interested reader is referred to the cited literature.

2.1. SACK (NewReno with SACK option)

Strictly speaking, SACK [5] is a TCP optional feature introduced the better to cope with multiple losses (i.e. losses that occur on the same transmitted cwnd). It resolves a basic problem of TCP Reno and NewReno [14], which would otherwise be unable to recover more than one TCP segment per RTT. The larger the average cwnd, the higher the benefits achieved, as multiple losses are obviously more frequent. In GEO satellite communications, this option may turn out to be particularly useful because large cwnd are required to fill the satellite pipe, due to long RTTs. However, its actual impact on performance depends on the ability of the TCP variant to keep the cwnd open in spite of long RTTs and concurrent traffic. In the following the SACK option is enabled on all TCP variants (the default in Linux). The term "SACK" is used in the paper to briefly denote TCP NewReno with SACK, i.e. the most widely adopted variant on wired networks.

2.2. Hybla

Hybla was conceived by the authors [8] with the primary aim of counteracting the performance deterioration caused by the long RTTs typical of satellite connections. It consists of a set of procedures which includes an enhancement of the standard congestion control algorithm, the mandatory adoption of the SACK policy, Hoe's channel bandwidth estimation, timestamps and packet spacing techniques. Hybla congestion control differs from NewReno as the additive increase is not fixed, but parameterized to the connection minimum RTT (the longer the minimum RTT value, the larger the additive increment of the cwnd). The aim is to speed up the cwnd growth rate, granting long RTT connections the same instantaneous segment transmission rate as a comparatively fast reference connection. This is particularly useful in the presence of concurrent cross traffic.

2.3. PEPsal

PEPsal is an integrated PEP scheme based on the TCP-splitting technique [11]. Following this principle, the TCP end-to-end connection is split into two different TCP connections. The first, on the wired leg from the source to the satellite gateway, adopts a standard TCP; the second, from the satellite gateway to the satellite receiver adopts TCP Hybla. The advantages of adopting TCP variant specifically designed for a satellite link are discussed in detail in the cited paper.

2.4. BIC

BIC-TCP (Binary Increase Congestion control-TCP) [6] uses a binary search algorithm to estimate the right *cwnd value*. This algorithm, which encompasses several phases, replaces the additive increase policy of NewReno. Without entering into details, the rationale is to allow faster growth of the cwnd when it is far from a target value derived from previous cwnd dynamics. In the presence of a loss, the multiplicative decrease policy of NewReno is preserved, although the cwnd is no longer halved (backoff factor 0.5) but only reduced (suggested backoff factor 0.8).

2.5. Vegas

Vegas [7] aims to prevent congestion losses by means of a dynamic estimate of the available bandwidth. This TCP variant computes the difference between the actual and the expected throughput and tries to keep it between two thresholds, α and β ($\alpha < \beta$), by adjusting (i.e. increasing, decreasing or leaving unchanged) the cwnd accordingly. By contrast with the standard loss-based congestion control algorithm, which "needs" a loss to reduce the cwnd, in the rate-based Vegas congestion control the cwnd can be reduced even in the absence of loss to prevent incoming congestion. Vegas also presents a number of minor modifications with respect to NewReno, included the use of timestamps to avoid spurious timeouts.

As Vegas relies on RTT variations to infer congestions (longer RTT are assumed to be indicators of longer queues in intermediate routers), this protocol is exposed to RTT variations due to other causes, as will be shown in the paper.

2.6. Westwood

TCP Westwood [10] was introduced to make the TCP congestion control more robust against losses not due to congestion, which are much more frequent in wireless than in wired links. To this end, Westwood introduces a modification of the Reno Fast Recovery algorithm called "Faster Recovery". Instead of halving the cwnd after three duplicate ACKs, and fixing the slow start threshold (ssthresh) at this value, TCP Westwood sets the ssthresh at an estimate of the available bandwidth. In this way, channel losses do not cause the dramatic slow-down of the transmission rate typical of the standard versions. The available bandwidth is estimated sender side, by monitoring the rate of returning ACKs. The original algorithm has been continuously updated by introducing a series of enhancements, including the Eligible Rate Estimator (ERE), the Adaptive Bandwidth Share Estimation (ABSE) [15]. So far TCP Westwood has been implemented in the Free BSD OS.

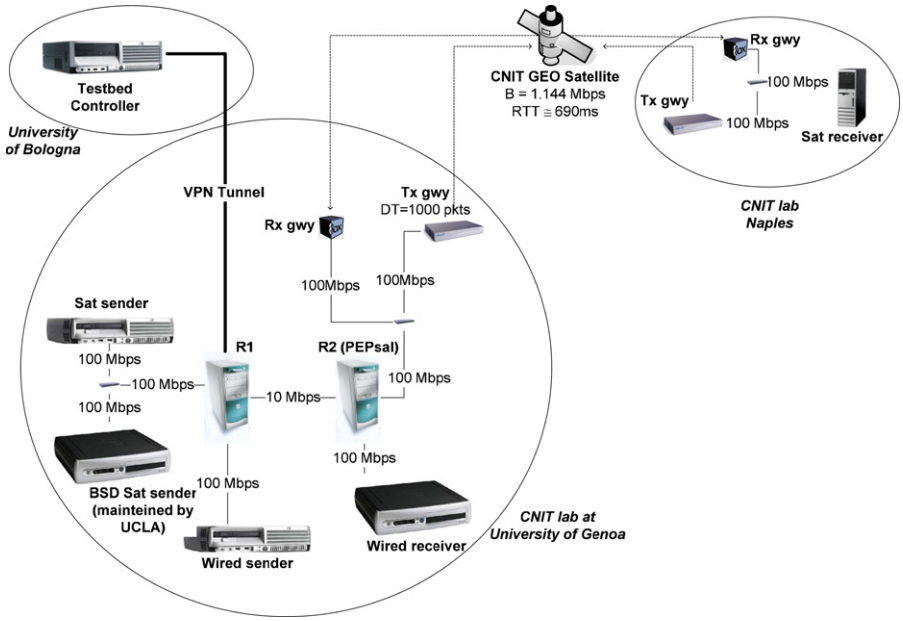


Fig. 2. Physical layout of the testbed actually used in the experiments.

2.7. Westwood+

Westwood+ [9] is a variant of the first Westwood version, proposed and developed in parallel by some of the Westwood authors. It introduces a slightly modified version of the original bandwidth estimation algorithm, in order to cope with ACK compression effects. Westwood+ is the only Westwood version available on Linux OS.

3. Testbed description

The logical layout shown in Fig. 1 is a simplified representation of the much more complex physical layout (Fig. 2) of the testbed actually used in our experiments. The real testbed consists of the following three geographical sections: the testbed controller located at UoB, the UoB-CNIT integrated testbed located at Genoa, and the Skyplex GEO satellite system whose receiver is located at the CNIT lab of Naples.

3.1. The testbed controller

The controller, connected to the testbed via a Virtual Private Network, allows the user ubiquitous remote control via a standard web browser. In addition to this important feature it provides the following advantages:

- to conceal the software and hardware complexity of the testbed;
- to improve security, as the user must be granted only standard access to the testbed web page without direct access to testbed components;
- to facilitate test scheduling especially when many users are using the testbed;
- to greatly speed up both test execution and result collection.

3.2. The integrated testbed

The UoB-CNIT integrated testbed consists of 5 Linux PCs and 1 FreeBSD PC (devoted to TCP Westwood and directly maintained by UCLA). Multi-TCP [16], installed on Linux senders, allows easy selection of the TCP variants available on Linux kernel 2.6.20 and collection of logs, and also implements the full version of TCP Hybla (including packet pacing and Hoe's initial ssthresh estimation).

Satellite connections are composed of wired legs (from the sat senders to R2, see Fig. 2) and a final satellite link (from R2 to the sat receiver), while TCP background traffic is present only in the entirely wired paths (from the wired sender to the wired receiver). All the connections share the R1–R2 link, whose bandwidth can be limited in order to study congestion effects. The router R1 can follow either a DT (Drop Tail) policy or a RED (Random Early Detection) policy. In both cases, the router parameters are fully configurable from the web interface. Finally, PEPsal is installed on the router R2.

3.3. The Skyplex platform

The Skyplex platform, which implements a mesh topology, is based on the Eutelsat HotBird 6 satellite, which carries four Ka band transponders equipped with a Skyplex Unit [1]. The Satellite Terminals (ST) are composed of channels configurable in Low Rate (at 2.112 Mbit/s) or in High Rate (at 6.226 Mbit/s). The channels can be configured in SCPC (Single Channel Per Carrier) or TDMA (Time Division Multiple Access) mode. Capacity sharing is available only in the latter case. The Skyplex TDMA mode has a *frame* structure composed of N time slots, each containing M MPEG-TS cells for a total of $N \times M$ cells per frame. N also represents the number of users per frame. In order to increase the number of users sharing the channel, the multiplexing structure has been extended by grouping L frames in a *multi-frame*. By assigning one time slot per multi-frame to each user, the maximum number of users per channel becomes $N \times L$. As in the satellite platform used in our tests $N = 6, L = 8$, and the channel is configured in Low Rate, we have a minimum granularity of 44 kbit/s. Finally, 3 multi-frames make up a *super-frame*, defined for signaling purposes. The super-frame time in our configuration lasts 820 ms, which represents the validity period of the schedule for the assignment of the multi-frame time slots to the terminals. Such a schedule is called the Burst Time Plan (BTP).

In the Skyplex platform, bandwidth assignment can be either dynamic (Bandwidth on Demand, BoD), or static (Committed Information Rate, CIR), or mixed. In the BoD case, the bandwidth request is calculated periodically by each terminal on the basis of its own instantaneous need, and the time slots are assigned in a best-effort mode. Neither priority between terminals nor quality of service can be guaranteed. In the case of static allocation, a fixed number of time slots is pre-configured and assigned a priori to each terminal. Our system adopts a mixed CIR/BoD assignment policy. Each active ST is given 44 kbit/s as static allocation, while the dynamic bandwidth component is assigned following a RBDC (Rate-Based Dynamic Capacity) [17] algorithm, which allocates bandwidth to users on the basis of traffic data rate registered on their network interfaces (i.e. on the Skyplex modem buffers). The bandwidth allocation procedure is performed by a Network Control Centre (NCC) upon reception of periodical requests generated by the STs. Each ST requests a bandwidth assignment proportional to the data rate of traffic incoming into its buffer. This strategy may pose severe challenges to TCP performance in consequence of both the inaccuracies of bandwidth assignment requests and the long allocation delays. The former are computed as proportional to the increase rate of the buffer queue, and are periodically updated every 820 ms (super-frame duration). In the case of Constant Bit Rate (CBR) traffic (e.g. UDP), this estimation process computes exact bandwidth requirement. By contrast, in the case of Variable Bit Rate (VBR) traffic (e.g. TCP) the estimation may be quite inaccurate, because of large fluctuations in the data rate. For instance, during the TCP slow start phase, the RBDC mechanism intrinsically underestimates the bandwidth actually necessary to accommodate the exponentially increasing traffic. As far as the allocation delay is concerned, this consists of three components: bandwidth request processing time (required by STs), bandwidth assignment processing time (required by the NCC) and the propagation delay intrinsic in two-way communication. This allocation delay, theoretically computed as 1.346 s in [17], reduces TCP congestion window growth rate, and contributes to TCP performance degradation [18]. Finally, it is worth noting that the bandwidth actually available is less than the 2.112 Mbit/s Low Rate because it depends on how many STs are on. During our experiments a total of 22 active STs reduced the nominal available bandwidth to 1.144 Mbit/s.

4. Evaluation tools and measurement objectives

In this section evaluation tools and performance metrics are described. They are functional to the description of the experimental results.

4.1. Evaluation tools

Adopting Multi-TCP [16] offers the additional advantage of overcoming the limitations of standard Linux evaluation tools, as it can collect from the kernel core a number of TCP variables, including the internal ones such as the cwnd and the retransmission timeout (RTO). Alternative choices for TCP dynamics analysis range from coarse tcpdump to TCP web100 instrumentation [19].

Data collection is accomplished through the insertion into the kernel TCP of three log routines, triggered by the following events:

- the encapsulation and passing of a TCP segment to the IP layer;
- the reception of a segment with the ACK flag active;
- the modification of the cwnd value.

The first routine reports the sequence number of the segment being sent out, its size, the total size of data sent (excluding retransmissions), and some variables related to the TCP pacing algorithm. The second gives the ACK number, the SND_UNA and SND_NXT values (as described in [20]), the estimated value of the "in flight" data, the receiver advertised window, the current state of the "Linux TCP/SACK/FACK/ECN engine state" [21] (OPEN, RECOVERY, etc.), and the three time variables, namely the mean RTT, the RTO, and the RTT variance. The last one prints out the parameters related to the congestion control algorithm, i.e. the cwnd, the ssthresh, and the maximum cwnd value (cwnd clamp).

Table 1
RTT characterization of the satellite link.

RTT	Min (ms)	Average (ms)	Max (ms)	Mean deviation (ms)
	587.35	692.5	802.75	58.92

All these data are collected in text files, which can be directly imported into a spreadsheet. Then, MS Excel macros specifically implemented in Visual Basic make the data processing phase fast and easy, by automatically generating a set of default graphs [16].

4.2. Performance metrics and measurement objectives

The aforementioned tools let us collect, at the sender side, information on transmitted/received packets, cwnd, etc. Starting from those data, the following metrics are considered in order to evaluate TCP performance:

- *time sequences* of transmitted packets and acknowledgments, useful for the study of both connection efficiency and regularity;
- *goodput*, defined as the average amount of correctly received data (excluding retransmissions) in the time unit; it is obtained at sender side by dividing the highest received ACK at the end of the connection by the connection length;
- *short time goodput*, defined as goodput, except that received ACK are sampled every second and averaged on a short time sliding window (10 s); it allows the study of the performance dynamics (e.g. at start-up, or in the presence of nonstationary concurrent traffic). Note that, during long recovery phases, cumulative acknowledgments can induce large and abrupt variations in short time goodput; to avoid these, SACK information has also been considered.

These metrics are used for the following measurement objectives (correspondent metrics are reported in brackets):

- evaluation for different file transfer lengths (goodput);
- evaluation of start-up performance (time sequences of packets and short time goodput);
- evaluation of the protocol ability to adapt to dynamic changes in network traffic (short time goodput);
- evaluation of fairness between satellite and wired connections on a heterogeneous network (short time goodput and goodput).

All results presented in the evaluation sections (with the exception of time sequences of Fig. 3) are averaged over many identical repetitions of each individual test, in order to cope with random effects introduced by the RED queue at bottleneck and, to a lesser extent, by the Skyplex BoD technique. Moreover, for the histograms presented in Figs. 4, 6 and 8, a 90% confidence interval is given.

5. Pure satellite network

This first scenario aims to evaluate the impact of satellite link characteristics on TCP performance, when the satellite connections are isolated from the other network components (i.e. in the absence of wired cross traffic).

5.1. Link characterization

A basic characterization of the satellite link is given in terms of RTT and segment loss rate. Table 1 shows some RTT measures obtained by using the "ping" [22] tool.

In order to probe the available bandwidth of the satellite link and the segment loss rate due to radio impairments, persistent (600 s) UDP flows were adopted. UDP traffic is regularly spaced (inter-packet delay is constant at the sender side) and does not react to losses. These important features allowed us to estimate available bandwidth simply by increasing the transmission rate and received counting packets. Measurements showed that UDP flows at 0.5 Mbit/s and 1 Mbit/s did not induce any loss, whereas 2 Mbit/s flow caused 48% of loss. The reasonable conclusion is that the physical layer error correcting code is able to assure a virtually loss-free satellite link and that the observed losses should be ascribed only to the bandwidth limitation on the satellite channel. Note that, in the absence of concurrent traffic, the maximum goodput of satellite connections is limited to this value. It is worth noting that the available net bandwidth (slightly more than 1 Mbit/s) is lower than the nominal value offered by the Skyplex System (1.144 Mbit/s, as pointed out in Section 2.3), due to the fact that data transmission is not performed continuously but in slot bursts, and moreover, the presence of guard times between consecutive slots and signaling reduces the available system bandwidth.

5.2. Single satellite connection

The aim of these tests is to assess start-up performance and goodput for a single satellite connection. In particular, the ability to exploit the available satellite channel bandwidth is measured by considering file transfers of different duration.

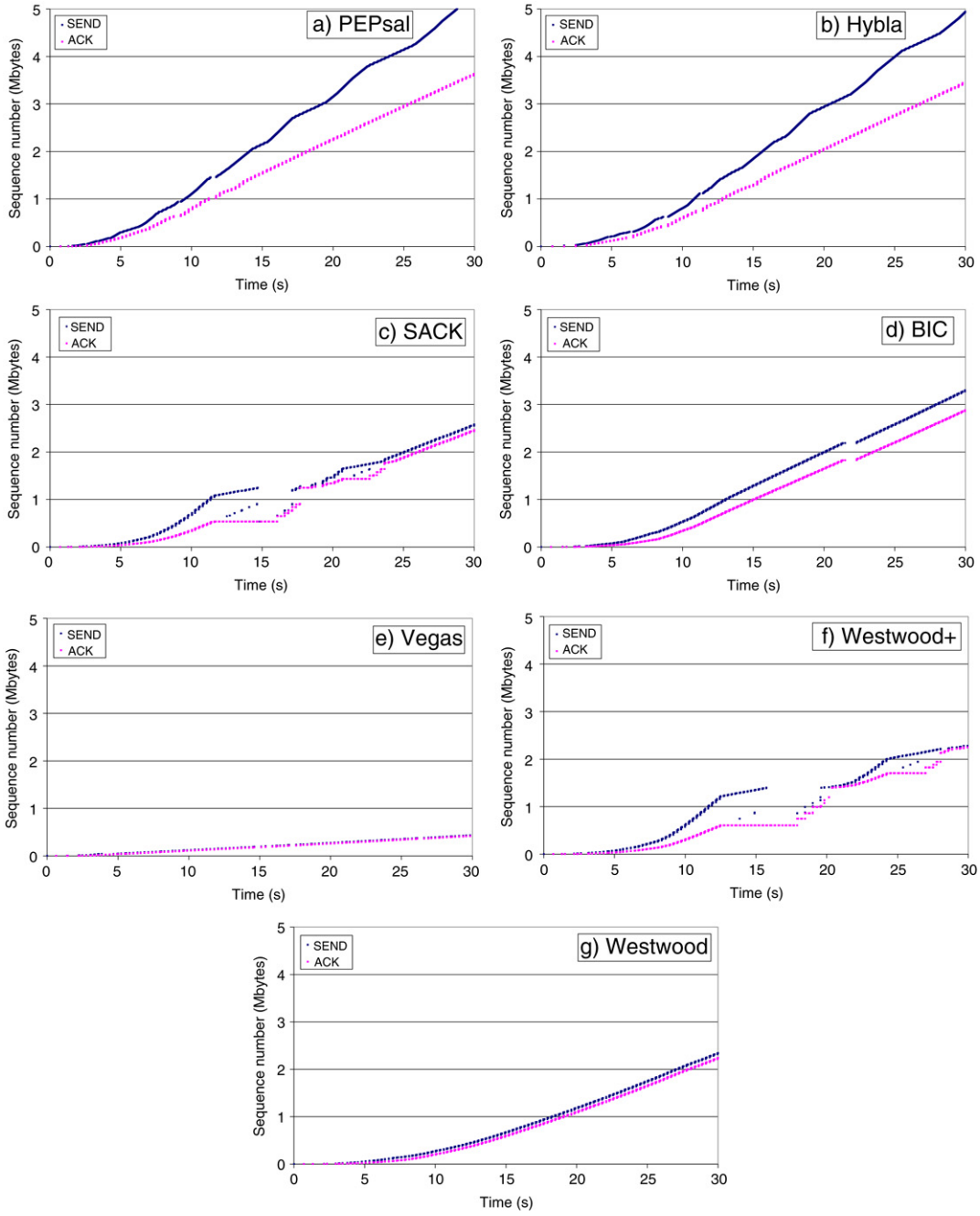


Fig. 3. Transmitted segments and acknowledgments at start-up: PEPsal (a), Hybla (b), SACK (c), BIC (d), Vegas (e), Westwood+ (f) and Westwood (g). Not averaged results.

Table 2 reports the time taken by each protocol to attain a goodput equal to 90% of link capacity taking into account long file transfers (about 180 s). All variants (but Vegas) are able ultimately to approach channel capacity although PEPsal and Hybla are the fastest (about 17 s against 29 s observed for SACK). Their performance is basically the same because PEPsal makes use of TCP Hybla in the satellite leg and the long RTT is the only limiting factor here.

In Fig. 3, time sequences of transmitted packets and acknowledgments are reported in order to explain the behavior at the start-up. As the sole exception to the general rule, results in this figure are not averaged, but refer to the best case. This is because time sequences cannot be averaged on multiple tests, by contrast with goodput results presented in all other figures. Let us start by considering PEPsal and Hybla time sequences (Fig. 3(a) and (b), respectively). PEPsal data refer to the satellite connection, which are the most significant for comparison purposes. They look similar, because Hybla is also adopted on the satellite leg of PEPsal. By comparing them with SACK results (Fig. 3(c)), the performance gain achieved by Hybla and PEPsal

Table 2
Time to reach the 90% of the satellite link capacity for a single satellite connection.

TCP variant	PEPsal	Hybla	SACK	BIC	Vegas	Westwood	Westwood+
Time (s)	16	17	29	24	Not reached	32	28

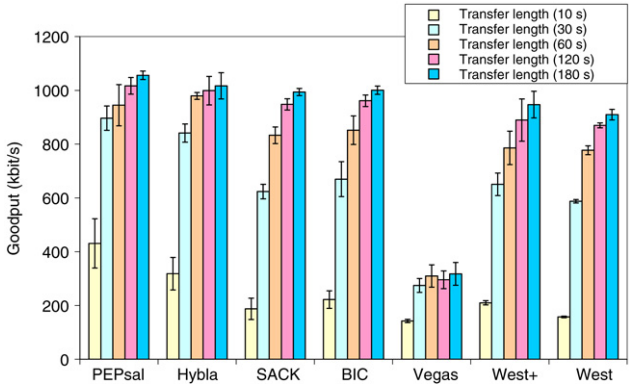


Fig. 4. Goodput of a satellite TCP connection for different transfer lengths. Averaged results and 90% confidence intervals.

is evident. This improvement is a consequence of the Hybla enhanced congestion control algorithm, specifically designed to counteract the effects of large RTTs. It allows a fast opening of the cwnd and, consequently, a fast increase in transmission speed. To this regard, it is worth noting that, by contrast with the other protocol variants, Hybla and PEPsal transmission rate is limited, during the initial slow start phase, by the start-up dynamics of the Linux receiver advertised window [23]. Moreover, the additional features of packet pacing and initial bandwidth estimation, allow Hybla to avoid the early buffer overflow at the first router (R1) that affects SACK (Fig. 3(c)) and Westwood+ (Fig. 3(f)), both of which give intermediate performance results. Moving to BIC (Fig. 3(d)) and Westwood (Fig. 3(g)), their good performance is due in both cases to a proper initial ssthresh value. However, while the former arbitrarily fixes the initial ssthresh at 100 packets [24], the latter exploits its Adaptive Bandwidth Share Estimation algorithm (ABSE) [15], which continuously updates the ssthresh value. It is worth noting that BIC's fixed value for the initial ssthresh could lead to suboptimal bandwidth exploitation in Fast Long-Distance Networks, even if it is a reasonable value in satellite environments. In spite of this advantage, both BIC and Westwood lack explicit countermeasures against long RTTs; this explains why their performance appears slightly worse than that of Hybla and PEPsal. Finally, Vegas (Fig. 3(e)) is greatly impaired by the satellite link characteristics and is unable to reach the target 90% of link capacity (Table 2). This behavior, observed throughout all the tests, is very likely due to a harmful interaction between the Skyplex BoD scheme, which increases the connection RTT variance [17], and the Vegas congestion estimation algorithm.

Another important remark is about the gap between transmitted packets and acknowledgment curves, particularly evident in PEPsal and Hybla time sequences (Fig. 3(a) and (b)). This gap is basically due to the presence of 1 MB buffer at the Skyplex satellite gateway (equivalent to 667 packets, with a packet length of 1500 bytes), which, being directly set by the satellite provider, was out of the user control. This is quite a large value, compared with the bandwidth delay product of the satellite link, which is of about 90 kbytes, and can cause a significant additional delay when the buffer queue becomes significant. This happens first for PEPsal and Hybla that, being faster, feed the satellite gateway buffer in a shorter time.

Finally, Fig. 4 shows the goodput of each variant considering different file transfer lengths. As highlighted above, all variants (but Vegas) succeed in exploiting the satellite link capacity in the case of long file transfers. By contrast, when short file transfers (e.g. 10 s) are considered, all variants (but PEPsal and Hybla) are severely impaired by the large RTT and the Skyplex BoD mechanism, which slows down the cwnd opening, in particular at start-up, confirming what was already observed in [17].

6. Heterogeneous network

This scenario represents a more realistic situation where satellite TCP connections are no longer isolated, but have to compete for bandwidth resources with wired connections. The aim is to evaluate the impact of wired cross traffic, either UDP or TCP, on a satellite connection.

6.1. UDP spike

This test considers a short (10 s) UDP traffic spike disturbance. One TCP satellite connection runs for 180 s, perturbed after about 60 s (i.e. after the start-up phase finishes) by a UDP flow at 9.5 Mbit/s on the wired path (cross traffic). In this

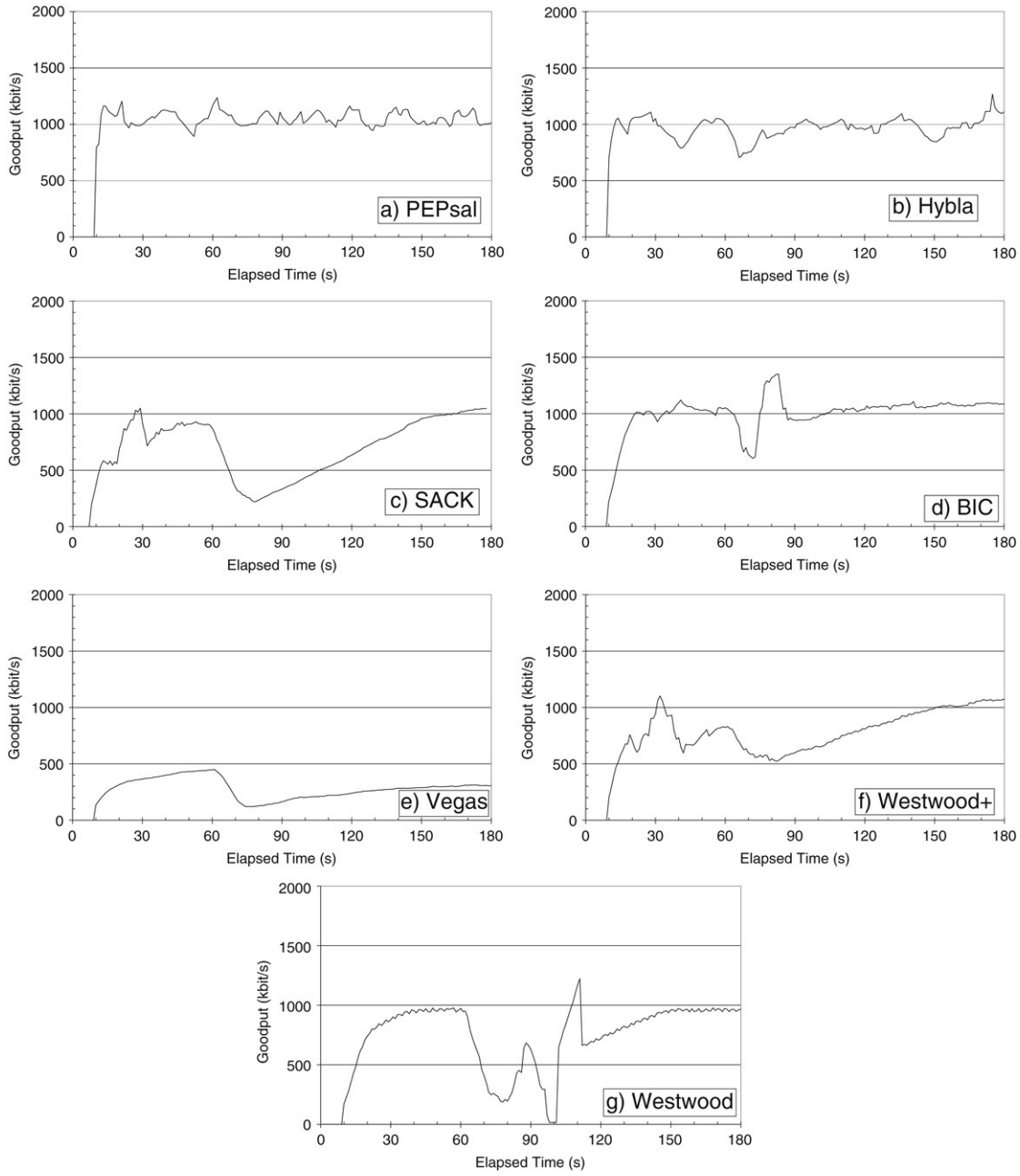


Fig. 5. Short time goodput in the presence of a UDP spike: PEPsal (a), Hybla (b), SACK (c), BIC (d), Vegas (e), Westwood+ (f) and Westwood (g). The UDP spike perturbation starts at 60 s and lasts 10 s. Averaged results.

way, the available bandwidth on the R1–R2 link is abruptly reduced for 10 s to only 0.5 Mbit/s, i.e. about half of the satellite bandwidth. Note that with this choice, during the UDP spike the bottleneck of the satellite connection shifts from the satellite link to the R1–R2 link. Given that UDP does not reduce its speed in the presence of congestion, the capability of the satellite connection to react to congestion-insensitive perturbation is the object of this test.

In Fig. 5 short time goodput graphs show clearly that PEPsal (Fig. 5(a)), Hybla (Fig. 5(b)) and BIC (Fig. 5(d)) are all very effective in coping with the UDP spike. However, the way they achieve this important result is different and deserves to be analyzed in some detail. PEPsal, by splitting the connection into two legs, isolates the wired leg, where the UDP spike is applied, from the long propagation delay of the satellite link. In this case, PEPsal's good performance is due more to the splitting concept itself, than on the use of an optimized TCP version on the satellite leg. By contrast, the good performance of Hybla has to be ascribed to the modified cwnd growth algorithm, which re-opens the cwnd very quickly at the end of the UDP perturbation. Finally, in the BIC case, the good performance is due to the binary search and multiplicative decrease

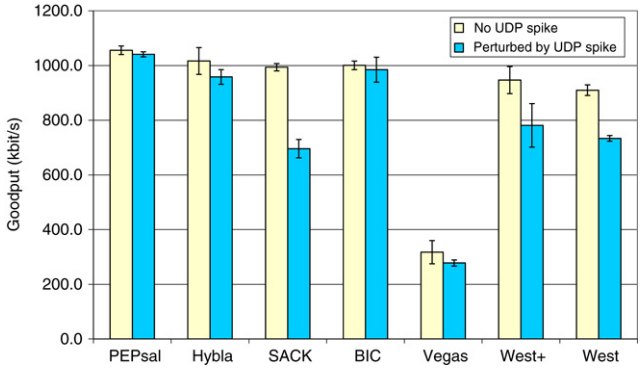


Fig. 6. Goodput penalization of a 180 s satellite TCP connection due to a UDP spike starting at 60 s and lasting for 10 s. Averaged results and 90% confidence intervals.

algorithm, which is fast in dynamically adapting the sending rate to the perceived network congestion. In all cases the capacity of the satellite gateway buffer helps to maintain a high transmission rate on the satellite link during the spike.

As regards SACK, Westwood, Westwood+ and Vegas, they are all significantly affected by the UDP spike. In particular, for SACK (Fig. 5(c)), about 90 s are necessary to restore the transmission speed achieved before the spike. The explanation of this unsatisfactory behavior is that during the UDP spike the cwnd is reduced to a very low value due to two consecutive recovery phases followed by a timeout. When the spike ends, the cwnd is increased in Congestion Avoidance (CA) phase of only one segment per RTT. As the RTT is quite long for the GEO satellite connection (more than 600 ms), the cwnd growth is extremely slow and consequently a very long time is necessary to recover from the perturbation. Regarding the remaining variants, it is worth noting that Westwood+ (Fig. 5(f)) is slow in reducing the transmission rate, but also in increasing it at the spike end, due to the standard CA algorithm. By contrast Westwood (Fig. 5(g)) seems to be much more reactive, both in reducing and increasing the transmission rate, owing to the ABSE algorithm. In these cases the satellite gateway buffer, as it is empty at the UDP spike start, cannot contribute in maintaining the transmission rate.

For a direct comparison of PEPsal and various TCP variants, it is convenient to consider goodput instead of short-term goodput as performance metrics. More precisely, in Fig. 6 goodput in the presence and in the absence of the UDP spike is plotted, in order to highlight the penalization due to the UDP traffic. PEPsal, Hybla and BIC ability to cope with the UDP spike is evident in contrast to the poor performance observed for SACK. Westwood and Westwood+ show intermediate results, in line with the short-term goodput analysis previously carried out. Finally, as regards Vegas, penalization is low, but the goodput itself, with and without the UDP spike, is always much lower than for the other tested variants.

6.2. TCP wired cross traffic

This series of tests investigates the behavior and the performance of a TCP satellite connection in the presence of congestion caused by concurrent TCP wired cross traffic. The aim is to assess the ability of different TCP variants at addressing the RTT penalization suffered by satellite traffic on a heterogeneous network because of the simultaneous presence of competing terrestrial traffic. As the behavior of some TCP variants may vary depending on the presence or otherwise of congestion at the TCP satellite connection start, we distinguish three possible cases: "wired cross traffic entering", in which the satellite connection starts before wired ones, "satellite entering", which is the inverse case, and "simultaneous insertion", in which all connections start at the same time.

6.2.1. Wired cross traffic entering

The objective of this test is to investigate the ability of the satellite connection to exploit the satellite channel in spite of competing cross traffic entering the network. To this end, the UDP spike previously considered is replaced by 5 persistent wired TCP connections (SACK, RTT $=$ 25 ms). They are launched 60 s after the satellite connection and last until the end of the test. Note that the sat capacity ($<$1.144 Mbit/s) is less than the maximum fair share on the R1–R2 link (1.67 Mbit/s), given by its bandwidth (10 Mbit/s) divided by the six competing connections. Therefore, by contrast to the UDP spike case, the satellite capacity can in principle be fully exploited, provided that the protocol is able to achieve the same capacity on the R1–R2 link. Fig. 7 shows that only PEPsal (Fig. 7(a)) reaches this objective, showing a short-term goodput completely insensitive to the wired traffic insertion. This excellent result is a direct consequence of the splitting technique. Indeed, the first leg of the satellite connection (from sat sender to R2) is characterized by the same short RTT and protocol (SACK) of the five competing wired connections, which resolves any RTT unfairness problem at the root. Moreover, considering the second leg (from R2 to sat receiver), there are neither other connections competing for bandwidth nor random losses (thanks to the Skyplex physical layer error correcting codes), therefore the available satellite bandwidth can easily be exploited. The adoption of Hybla on the satellite leg provides only a limited advantage here. However, its speed in cwnd opening would have been advantageous when dealing with shorter file transfers (see Fig. 4) and crucial in the presence of channel errors [11].

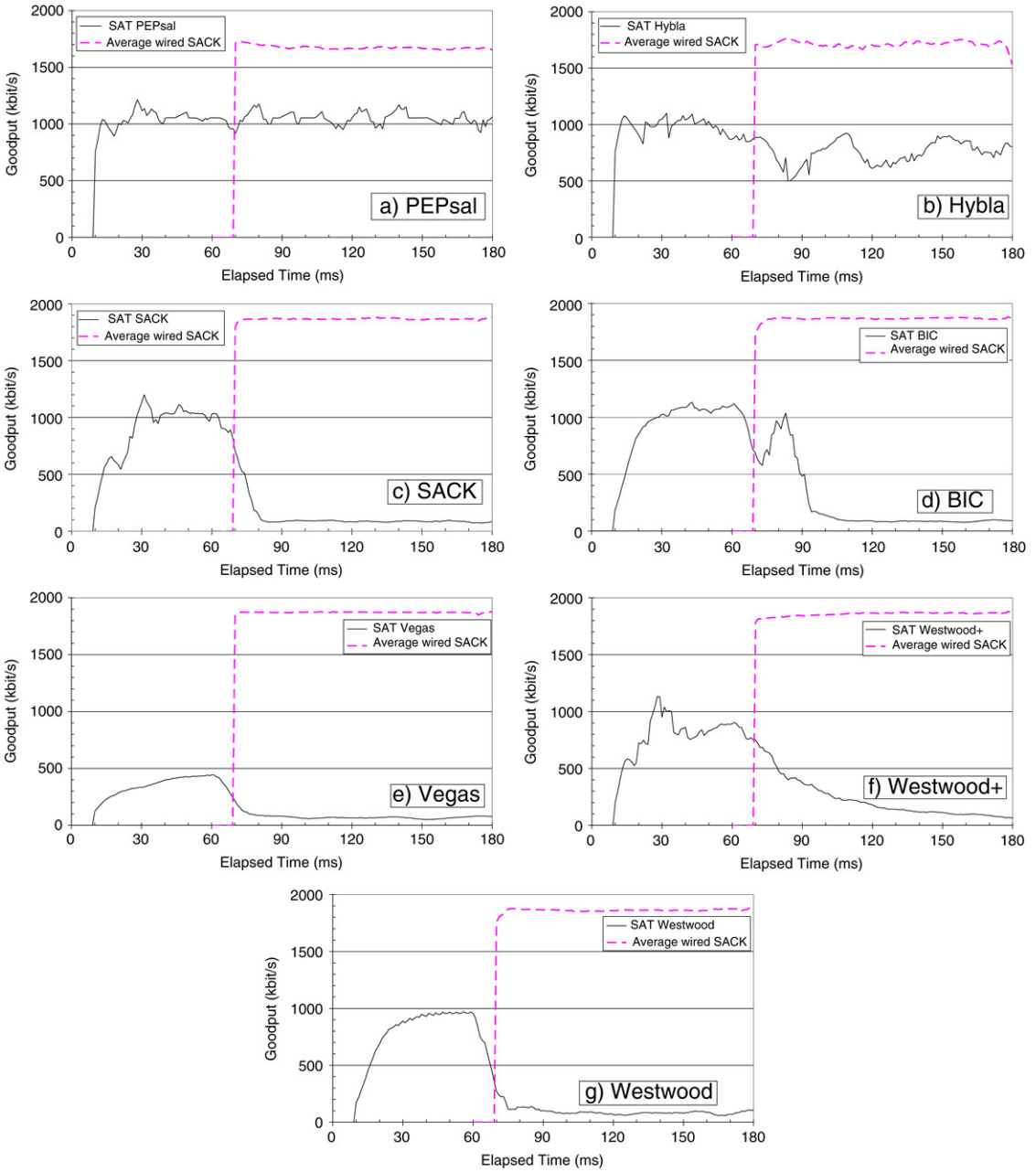


Fig. 7. Short time goodput of a satellite connection in the case of wired cross traffic arriving: PEPsal (a), Hybla (b), SACK (c), BIC (d), Vegas (e), Westwood+ (f) and Westwood (g). Wired connections start 60 s after sat connection. Averaged results.

As far as the end-to-end TCP variants are concerned, Hybla (Fig. 7(b)) achieves by far the best performance, although, by contrast with PEPsal, it is clearly affected by the wired traffic insertion and the final transmission speed results lower than the satellite capacity. This performance disparity is due to the different loss recovery time, which in Hybla (as well as in all end-to-end TCP variants) is related to the end-to-end RTT, while in PEPsal it is related to the much shorter RTT of the wired leg.

As regards SACK, Vegas and Westwood, (Fig. 7(c), (e) and (g), respectively), the lack of any explicit countermeasures against RTT unfairness prevents them from maintaining an adequate share of network resources after the insertion of wired traffic. The same comments also hold for BIC (Fig. 7(d)) and Westwood+ (Fig. 7(f)), although they show a better resistance against cwnd shrinking and transmission speed reduction, after congestion event detection. This is due to their nonstandard cwnd reduction policy after a loss event, i.e. $cwnd = Eligible\ Rate\ Estimate$ on Westwood+ and $cwnd = 0.8 \cdot previous\_cwnd$ on BIC. However, they eventually converge to a short-term goodput value much lower than the satellite channel capacity.

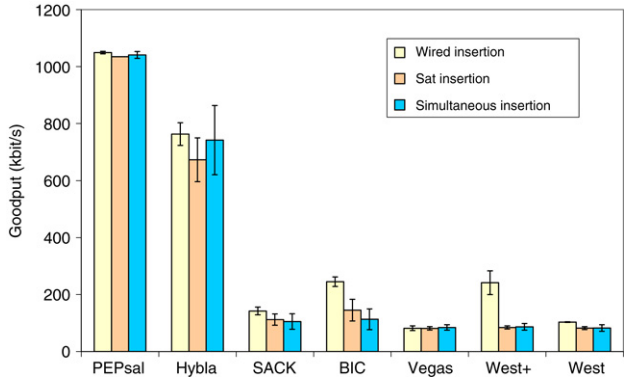


Fig. 8. Goodput of a 180 s satellite TCP connection in the presence of wired cross traffic: wired insertion, sat insertion, and simultaneous insertion. Averaged results and 90% confidence intervals.

A more direct comparison can be made by considering goodput instead of short-term goodput as a performance metric. To this end, the satellite connection goodput for all the solutions examined is given in Fig. 8. Note that the picture presents not only data referring to the wired cross traffic entering case, considered here, but also the other two cases (namely, satellite entering and simultaneous insertion), which we examine below.

6.2.2. Satellite entering

At time zero, five wired connections (SACK, RTT=25 ms) are launched, followed, after about 60 s, by the satellite connection. In this case the satellite connection is established when the network is already congested, i.e. from the most unfavorable condition. The results (Fig. 8) confirm the predominance of PEPsal, which is the only one able to fully exploit satellite capacity. Hybla follows immediately, while all the other variants are able to use just a fraction of the satellite bandwidth. Comparing the previous outcomes with those obtained here, it can be noted that BIC and Westwood+ are more suitable to preserve the achieved transmission rate when cross traffic appears, than to increase it in the case of already existing cross traffic.

6.2.3. Simultaneous insertion

Finally, we consider the intermediate case of simultaneous start at time zero of both the sat connection and the five competing wired connections. The result trend (Fig. 8) is very similar to the previous case; this behavior is expected if we consider that, thanks to their shorter RTT and the lack of the Skyplex BoD mechanism penalization, wired connections are much faster than the satellite connection in increasing transmission rate. Note, in this regard, that by contrast with other TCP variants, Hybla start-up performance is limited not by an insufficient cwnd growth rate, but by Linux advertising window dynamics (as well as by the BoD penalization), as previously pointed out. The same holds true for PEPsal. However, in this latter case, thanks to the splitting concept, the same Hybla protocol applied on the satellite leg does not have to directly compete with short RTT wired connections, thus making PEPsal performance invariant in all the considered cases.

7. Conclusions

In this paper, the outcomes of a live measurement campaign on a real satellite platform are reported. The results confirm the severity of the challenges posed by real GEO satellite channels, as well as the impact of BoD techniques on TCP start-up performance. PEPsal, a TCP splitting PEP implementation developed by the authors, achieved the best performance in all the tests, being able to fully exploit the satellite channel capacity owing to its ability to isolate satellite channel impairment effects. However, the splitting approach it follows has the drawbacks of violating end-to-end semantics and preventing the use of IPsec protocol. Whenever services provided by satellite operators are not compatible with these restrictions, an end-to-end TCP connection must be used. Among tested end-to-end TCP variants, Hybla provides the best results, especially in heterogeneous environments. Further research will extend comparison to other satellite platforms and/or different environments, as well as to new TCP enhancements or other transport layer solutions.

Acknowledgements

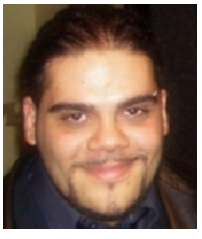
This work was supported in part by the IST-027323 SatNEx II project (NoE), funded by the European Community within FP6 program.

References

[1] E. Feltrin, E. Weller, E. Martin, K. Zamani, Design, implementation and performance analysis of an on board processor-based satellite network, in: Proc. of International Conference on Communications June 2004, vol. 6, pp. 3321–3325.
[2] Y. Hu, V.O.H. Li, Satellite-based internet: A tutorial, IEEE Commun. Mag 39 (3) (2001) 164–171.
[3] C. Caini, R. Firrincieli, M. Marchese, T. de Cola, M. Luglio, C. Roseti, N. Celandroni, F. Potortì, Transport layer protocols and architectures for satellite networks, Wiley Int. J. Satellite Commun. Netw. 25 (1) (2007) 1–26.
[4] J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby, Performance enhancing proxies intended to mitigate link-related degradations, IETF RFC, June 3135, 2001.
[5] M. Mathis, J. Mahdavi, TCP selective acknowledgment options, IETF RFC, 2018, Oct. 1996.
[6] L. Xu, K. Harfoush, I. Rhee, Binary increase congestion control for fast long distance networks, in: Proc. IEEE INFOCOM, Hong Kong, March 2004, vol. 4, pp. 2514 – 2524.
[7] L.S. Brakmo, L.L. Peterson, TCP Vegas: End to end congestion avoidance on a global internet, IEEE J. Select. Areas Commun. 13 (1995) 1465–1480.
[8] C. Caini, R. Firrincieli, TCP hybla: A TCP enhancement for heterogeneous networks, Wiley Int. J. Satellite Commun. Netw. 22 (2004) 547–566.
[9] L.A. Grieco, S. Mascolo, Performance evaluation and comparison of Westwood+, New Reno, and Vegas TCP congestion control, ACM Computer Commun. Review 34 (2004) 25–38.
[10] C. Casetti, M. Gerla, S. Mascolo, M.Y. Sanadidi, R. Wang, TCP Westwood: End-to-end congestion control for wired/wireless networks, Wireless Networks 8 (2002) 467–479.
[11] C. Caini, R. Firrincieli, D. Lacamera, PEPsal: A Performance Enhancing Proxy for TCP satellite connections, in: Interworking and Resource Management in Satellite Systems Special Series, IEEE Aerospace and Electronic Systems Magazine 22 (8) (2007) B9–B16.
[12] A. Bon, C. Caini, T. De Cola, R. Firrincieli, D. Lacamera, M. Marchese, An integrated testbed for wireless advanced transport protocols and architectures, in: Proc. IEEE TridentCom, Barcelona, Spain, Mar. 2006, pp. 522-525.
[13] Web site: https://tatpa.deis.unibo.it.
[14] S. Floyd, T. Henderson, A. Gurtov, The NewReno modification to TCP's fast recovery algorithm, IETF RFC, 3782, 2004.
[15] M. Gerla, B. Kwok Fai Ng, M.Y. Sanadidi, M. Valla, R. Wang, TCP Westwood with adaptive bandwidth estimation to improve efficiency/friendliness tradeoffs, Elsevier Computer Commun. 27 (1) (2004) 41–58.
[16] C. Caini, R. Firrincieli, D. Lacamera, A linux based multi TCP implementation for experimental evaluation of TCP enhancements, in: Proc. SCS SPECTS, Philadelphia, July 2005, pp. 875–883.
[17] A. Gotta, R. Secchi, F. Potortì, An analysis of TCP startup over an experimental DVB-RCS platform, in: Proc. IEEE IWSSC, Madrid, Spain, Sep. 2006, pp. 176–180.
[18] M. Sooriyabandara, G. Fairhurst, Dynamics of TCP over BoD satellite networks, Wiley Int. J. Satellite Commun. Netw. 21 (2003) 427–449.
[19] M. Mathis, J. Heffner, R. Reddy, Web100: Extended TCP instrumentation for research, education and diagnosis, ACM Computer Commun. Review 33 (3) (2003).
[20] J. Postel, Transmission control protocol, IETF RFC, 793, Sep. 1981.
[21] P. Sarolahti, A. Kuznetsov, Congestion control in linux TCP, in; Proc. USENIX Annual Technical Conference, Freenix Track, Monterey, CA, Jun. 2002, pp. 49–62.
[22] D.L. Mills, Internet delay experiments, IETF RFC, 889, Dec. 1983.
[23] J. Semke, J. Mahdavi, M. Mathis, Automatic TCP buffer tuning, ACM Computer Commun. Review 28 (4) (1998) 315–323.
[24] Y.T. Li, D.J. Leith, BicTCP implementation in Linux kernels, Hamilton Institute Technical Report, 2004 www.hamilton.ie/net/LinuxBicTCP.pdf.

C. Caini received the Dr. Ing. Degree (summa cum laude) in Electronic Engineering from the University of Bologna, Italy, in 1986. Since 1990, he has been with the Department of Electronics Computer Science and Systems of the same University, where he is currently an Associate Professor. His main scientific interests are in the field of terrestrial and satellite cellular mobile radio systems, with a special emphasis on spectrum efficiency, multiple access techniques and spread spectrum systems. The recent integration of Internet and wireless communications has led him to devote his recent research activity to the development of network protocols and architecture for satellite and wireless applications. He participates to several international research projects and he is author of many international publications on these topics. He is member of IEEE Communications Society.

R. Firrincieli is a senior researcher at the Advanced Research Center on Electronics Systems for Information and Communication Technologies (ARCES), University of Bologna, Italy. He received his Masters and Ph.D. degrees in Telecommunications Engineering from the University of Bologna in 2001 and 2006 respectively. During 2005, 2006 and 2007 he spent 10 months as Visiting Researcher at Department of Computer Science, the Henry Samueli School of Engineering and Applied Sciences, UCLA, US. His present interests involve the study and the evaluation of enhanced transport protocols, Performance Enhancing Proxies solutions, and Delay/Disruption Tolerant Networks architectures over wireless network. Moreover, he is interested in congestion control algorithms for unicast and multicast protocols, traffic shaping, retransmission techniques (ARQ, Data Carousel), and packet coding at the transport layer. He is co-author of 30 scientific publications on International Journals and Conferences.



D. Lacamera received a Bachelor Degree in Computer Science Engineering and a Master in Free/Open Source Software Technologies from University of Bologna. His main areas of interest are network protocols, TCP enhancements and operating systems implementation. He has been cooperating with DEIS research team since 2004. He implemented the TCP Hybla protocol in the Linux Kernel. He also developed several tools to measure TCP performance in UNIX and a free TCP Performance Enhancing Proxy (PEPsal). His further areas of interest are Embedded Devices, Virtual Machines and Virtual Networks. He participates to some free software projects, like the Virtual Distributed Ethernet which is developed at the University of Bologna. He is currently employed as senior software developer at Sadel S.p.A.

T. de Cola was born in Manosque, France, on April 28, 1977. He received the "Laurea" degree (summa cum laude) in telecommunication Engineering from the University of Genoa, Genoa, Italy, in 2001 and the Qualification degree as Professional Engineer in 2002.

From 2002 until 2007, he has worked with the Italian Consortium of Telecommunications (CNIT), University of Genoa Research Unit, as scientist researcher. Since 2008, he has been with the German Aerospace Centre (DLR), where he is involved in different European Projects focusing on different aspects of DVB standards, CCSDS protocols and testbed design. He is co-author of more than 20 papers, including international conferences and journals.

His main research activity concerns: TCP/IP protocols, satellite networks, transport protocols for wireless links, interplanetary networks as well as delay tolerant networks.

M. Marchese (IEEE S'94–M'97–SM'04) was born in Genoa, Italy in 1967. He got his "Laurea" degree cum laude in 1992, the Qualification as Professional Engineer in April 1992 and his Ph.D. in 1996, at the University of Genoa, Italy. From 1999 to 2004, he was Head of Research in the Italian Consortium of Telecommunications (CNIT). From February 2005 he has been Associate Professor at the University of Genoa, Department of Communication, Computer and Systems Science (DIST) where he is the founder and technical responsible of the Satellite Communications and Networking Laboratory (SCNL). He was an Officer of IEEE ComSoc Satellite and Space Communications Technical Committee since July 2002 (2002–2004: Secretary; 2004–2006: Vice-Chair; 2006-2008: Chair) where he is now Past-Chair. He was the Official Representative of CNIT within the European Telecommunications Standard Institute (ETSI) from 1999 to 2005.

He is author and co-author of more than 150 scientific works, including international magazines, international conferences and book chapters. He is the author of the book "Quality of Service over Heterogeneous Networks", John Wiley & Sons, Chichester, 2007. He is Associate Editor of the International Journal of Communication Systems — IJCS (Wiley) and of the IEEE Wireless Communications Magazine (WCM). He has organized Special Issues of International Journals and Magazines (Wiley IJCS, IEEE WCM, IEEE JSAC, Elsevier Computers and Electrical Engineering Journal, IEEE Systems Journal) and is Technical Committee Co-Chair of many international conferences including SPECTS, IEEE Globecom, IEEE ICC.

His main research activity concerns: Satellite and Radio Networks, Transport Architectures for Cable, Satellite and Wireless Networks, Quality of Service over Heterogeneous Networks, Performance Evaluation of Telecommunication Networks.

C. Marcondes is currently Computer Science Professor at UFSCar — Federal University of São Carlos, Brazil. He received Computer Science degree from State University of Londrina, Brazil in 1997, M.S. degree in Informatics from Federal University of Rio de Janeiro in 2002 and Ph.D. in Computer Science from UCLA in 2008. Dr. Cesar has extensive experience on research, in particular, in the areas of congestion control, network path estimators and IP telephony.

M. Y. Sanadidi was born in Egypt where he received his high school diploma from College Saint Marc D'Alexandrie, and his B.Sc. from the University of Alexandria. He earned his M.Sc. from Penn State, and his Ph.D. in Computer Science from UCLA.

He is currently an Adjunct Professor at the UCLA Computer Science Department, where he is co-Principal Investigator on NSF sponsored research in high performance Internet protocols. At UCLA, he also teaches undergraduate and graduate courses on computer networks, queuing systems, and probability modeling and analysis.

Dr. Sanadidi was a Manager and Senior Consulting Engineer at AT&T/Teradata, and previous to that, he held the position of Computer Scientist at Citicorp, where he pursued R&D projects in wireless metropolitan area data communications. In particular, he lead the design and prototyping of a Wireless MAN for home banking and credit card verification applications. Dr. Sanadidi was also an Assistant Professor at the Computer Science Department, University of Maryland, College Park, Md. There, he taught performance modeling, computer architecture and operating systems, and was Principal Investigator for NSA sponsored research in global data communications networks.

Professor Sanadidi, who is a Senior Member of the IEEE and Member of the ACM, has co-authored over forty conference and journal papers and has been awarded two best paper awards. He has served as reviewer of journal publications, member of technical program committees and organizing committees, keynote speaker, as well as chairman of professional conferences. Dr. Sanadidi has also been awarded two patents, and has consulted for a number of industrial concerns.

His current research interests are in path characteristics estimation and its applications in congestion control, adaptive multimedia streaming and other hybrid wire/wireless paths issues.

M. Gerla, Professor, UCLA, Computer Science Dept. Dr. Gerla received his Engineering degree from the Politecnico di Milano, Italy, in 1966 and the M.S. and Ph.D. degrees from UCLA in 1970 and 1973. He became IEEE Fellow in 2002. At UCLA, he was part of a small team that developed the early ARPANET protocols under the guidance of Prof. Leonard Kleinrock. He worked at Network Analysis Corporation, New York, from 1973 to 1976, transferring the ARPANET technology to several Government and Commercial Networks. He joined the Faculty of the Computer Science Department at UCLA in 1976, where he is now Professor. At UCLA he has designed and implemented some of the most popular and cited network protocols for ad hoc wireless networks including distributed clustering, multicast (ODMRP and CODECast) and transport (TCP Westwood) under DARPA and NSF grants. He has lead the $12M, 6 year ONR MINUTEMAN project, designing the next generation scalable airborne Internet for tactical and homeland defense scenarios. He is now leading two advanced wireless network projects under ARMY and IBM funding. In the commercial network scenario, with NSF and Industry sponsorship, he has led the development of vehicular communications for safe navigation, urban sensing and location awareness. A parallel research activity covers personal P2P communications including cooperative, networked medical monitoring (see www.cs.ucla.edu/NRL for recent publications).