An Analysis of TCP Startup over an Experimental DVB-RCS Platform

Alberto Gotta

DIST-University of Genoa and CNIT
University of Genoa Research Unit,
Via Opera Pia 13, 16145 Genoa (Italy)
alberto.gotta@isti.cnr.it

Francesco Potortì, Raffaello Secchi

Information Science and Technology Institute "Alessandro Faedo",
Italian National Research Council (C.N.R.),
Via G. Moruzzi, 1, San Cataldo, I-56124 Pisa (Italy)
{potorti,raffaello.secchi}@isti.cnr.it

***Abstract*— Satellite systems are evolving towards higher available bandwidths and dynamic allocation based on instantaneous traffic rates offered at the stations, so called BoD (bandwidth on demand) channel sharing. This trend is coupled with more and more powerful error correcting schemes, like those adopted in the recent DVB-S2 standard, which promise to make the channel virtually immune from packet errors. These factors combine so that most TCP connections would send all of their data during the Slow Start phase. We investigate the performance of TCP during startup on recent BoD system by observing and explaining the behavior of different TCP flavors on different systems when transmitting data over a Skyplex satellite system. We make recommendations for choosing and improving TCP implementations and for future BoD allocation schemes.**

I. Introduction

In order to improve TCP performance over satellite channels, many aspects of networking should be investigated. Data link protocols, application protocols, router buffer size, queuing disciplines and proxy location are just some of the issues to consider when facing the design of TCP network over satellite links [1]. Nevertheless, we show that simple mechanisms, such as that enforced by TCP stack implementations of FreeBSD kernels, may have an overwhelming impact on overall networks performance.

Recent and future satellite systems are characterized by quasi-ideal loss characteristics and broadband links. Consequently, in the first place congestion - as opposed to packet loss due to link errors - dominates the TCP dynamics, and in the second place the delay-bandwidth product is very large. As a consequence of these two effects, TCP sessions are typically concluded within the slow start phase, without incurring any packet loss.

The performance of the slow start phase is then extremely critical as far as network performance is concerned, but this issue has not received adequate attention from the research community, especially concerning experimental measurements on recent platforms. Moreover, the slow start phase is slowed down in BoD (Bandwidth on Demand) systems, which are currently emerging in the market.

In fact, the bandwidth is assigned on the basis of the stations' requests, which in turn depend on the current transmission rate and possibly of transmission backlog of each station. Requests from the stations need 250 ms propagation delay on geostationary satellite networks, and the allocation needs 250 ms to be broadcasted to the stations, meaning that assignments are always late with respect to incoming traffic by at least 500 ms, to which management overheads should be added. Since the throughput in the slow-start phase of TCP increases at each RTT, the allocated bandwidth is always less than the offered traffic, which accounts for the very long slow-start phase we observed in BoD systems.

In [2] we have measured and compared the improvement introduced by TCP variants and options (TCP Westwood and SACK option) in recent Linux kernels, by means of experiments on Skyplex, a commercial DVB-RCS satellite platform. [3] [4] do a simulative analysis of TCP behavior on BoD satellite systems. In this work we complement that analysis with measurement focusing on the startup behavior of Linux 2.6 (with and without SACK option) and FreeBSD implementations of NewReno TCP on the Skyplex platform.

II. The Skyplex platform

The measurements reported in this paper are carried out in an experimental satellite network based on Skyplex technology by Eutelsat [5], which relies on a subset of DVB-RCS features. In Skyplex, IP packets are inserted into a Mpeg-2 transport stream and transmitted to the satellite through Time Division Multiple Access (TDMA) uplinks using DVB-RCS format from small traffic terminals. On board of the satellite (HotBird 6), these signals are demodulated, regenerated and forwarded in the downlink in DVB-S format.

The HotBird 6 satellite is equipped with four Ka band transponders, whose footprint covers the most of the european area. The control and management is centralized: the control center, located in Lario (Italy), is in charge to generate messages sent to the traffic terminals used for acquisition and synchronization, and compute and broadcast the burst time plan (BTP) for the assignment of the capacity to the terminals.

In order to meet user requirements, the satellite bandwidth (about 36 Mbps) can be divided in Low Rate channels (LR at 2.112 Mbps) and High Rate channels (HR at 6.226 Mbps), having each channel a TDMA structure that allows a configurable number of time slot per frame. Our experimental testbed

consists of an LR carrier with a TDMA frame of 48 time slot.

The slot assignment is dynamic (BoD). The bandwidth is requested periodically by each terminal to the network control center on the basis of its own instant need, and the time slot are assigned in a best-effort mode. However, a minimum of guaranteed bandwidth equivalent to one slot per frame (44 kbps), used for signalling and user data, is reserved to each traffic terminal.

III. TCP Measurements over Skyplex

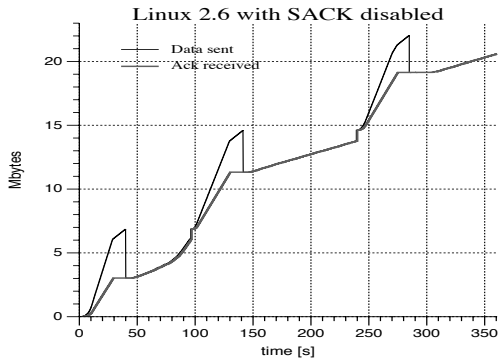
First experiments with Linux reveal that, on the first connection to a given destination, performance is severely hindered because of the first congestion event at the end of the slow-start phase, which causes the loss of several segments that TCP takes a long time to recover. Performance on subsequent connections is good if the cached value of the slow start threshold has not yet expired.

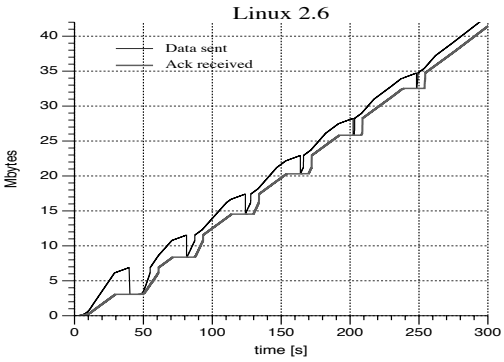
Instead, experiments with FreeBSD show good performance on all connections, because recent FreeBSD kernels perform an estimation of the bandwidth-delay product and prevent the congestion window from exceeding the estimated value, and completely prevent packet loss, at least in our setting. This method produces a slight underutilization of the satellite channel, while providing excellent startup performance even on the first connection. This behavior is highlighted in the figures, which show how Linux TCP, with and without SACK option, and FreeBSD TCP behave on a Skyplex link.

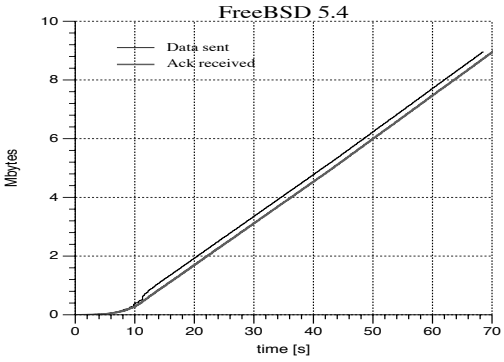
It is apparent from the figures that the inner workings and the overall performance of these three TCP flavors differ substantially:

- Linux without Sack fills up the bottleneck buffer (during phase 2) and experiments many packet losses when the buffer overflows; the subsequent fast recovery phase is not fast enough to avoid a timeout (phase 4) and a very slow Slow Start phase, that retransmits a high number of packets already transmitted, whose duplicate ACKs do not contribute to increasing the congestion window.
- Linux with Sack behaves much better, essentially because the Slow Start phase is fast thanks to the selective acknowledgments; the resulting throughput is, on average, as fast as the channel permits, even if the flow of packets is very irregular.
- FreeBSD uses an algorithm for estimating the bandwidth available to the TCP connection and the latency of the channel, and uses these estimates to cap the rate of packet transmission; the resulting behavior, in this simple case where a single connection occupies the whole channel, is almost as efficient on average as Linux and in addition the packet rate is extremely regular, without even a lost packet and with an RTT only slightly higher than the minimum.

We made several more experiments and we are able to explain why the behaviors are different, and to make recommendation to increase the performance of the various TCP flavors or recommend one. We also plan to make experiments on different platforms, such as Mac OS and Windows, and compare them with those we already have, in order to offer a wide view on the current TCP flavors available to the users.







References

[1] M. Allman, D. Glover, and L. Sanchez. RFC 2488: Enhancing TCP Over Satellite Channels using Standard Mechanisms, January 1999.

[2] N. Celandroni, F. Davoli, E. Ferro and A. Gotta. TCP performace measured over wireless integrated networks with high delay-bandwidth products. In *proc. of ASMS 2006*, May 2006. Herrsching am Ammersee, Germany.

[3] M. Sooriyabandara and G. Fairhurst. Dynamics of TCP over BoD satellite networks. *Int. Journal of Satellite Comunication and Networking*, 21(4-5):427–449, Jul 2003.

[4] M. Karaliopoulos, R. Tafazolli and B. Evans. On the interaction of TCP with BoD in GEO broadband satellite networks. In *proc. IEEE Globecom 2002*, November 2002. Taipei, Taiwan.

[5] E. Feltrin, E. Weller, E. Martin and K. Zamani. Design, Implementation and Performance Analysis of an On Board Processor-Based Satellite Network. In *ICC04*, June 2004.