

Internet via Satellite: Problems and Solutions

A Flash Networks Ltd. White Paper

**Michael Orr
Chief Technology Officer**

Internet via Satellite: Problems and Solutions

ABSTRACT

Satellite links are increasingly used for Internet/Intranet applications, which are predominantly based on TCP. Unfortunately, TCP exhibits marginal performance and excessive resource consumption in a satellite environment.

The causes for TCP's performance issues are presented, as well as some issues related to the usage of HTTP over satellite links, and the implications of carrying a large number of concurrent connections over a satellite link.

Various solutions are surveyed, and it is shown that throughput gains of 100%-500% and bandwidth savings of 30%-50% are achievable.

It is illustrated that these solutions may be integrated into existing networks without requiring a wholesale replacement of TCP in the end nodes, and a cost justification model is presented, illustrating the practicality and feasibility of these solutions.

SatBooster™, a Flash Networks Ltd. product improving Internet/Intranet performance and bandwidth usage over satellite links is briefly presented and used to demonstrate various points and figures.

Introduction

For over a decade, satellite networks have been used for data communication. The typical use of these networks traditionally tended toward Transaction based applications, where typically each transaction involved the exchange of very few packets over the network.

As applications shift toward an Internet/Intranet access model, each "Transaction" involves the exchange of a significant amount of packets. Under these conditions, "response time" becomes a measure of the time it takes to complete the entire exchange, and therefore is really a measure of Throughput.

Since traffic in the Internet/Intranet applications can be an order of magnitude larger than in the original transaction based network design, the question of resource utilization and requirements becomes a key consideration.

TCP's Performance over Satellite Links

As Internet/Intranet applications are heavily based on TCP as the underlying transport protocol, its performance profile over satellite links is of paramount importance.

Accumulated experience with TCP usage over satellite links shows that the performance and resource consumption of TCP over satellite links can be unacceptable [1, 6, 9, 10, 11].

We will examine separately issues limiting Throughput (i.e. Transfer speed) and Resource consumption.

Transfer Speed limiting Issues

Transmission rate setting

When a TCP connection is started, a built-in mechanism called "Slow Start" is used to find out the available link capacity. [12]. This algorithm is designed to cause TCP to start at a slow rate and then double the transmission rate for every acknowledgement (ACK) received. If this process is continued long enough, eventually the link capacity is exceeded, and at least one packet is lost. At that point TCP reduces the transmission rate back to the last rate known to be acceptable, which is 50% of the last attempted one. TCP then shifts into a mode called "congestion avoidance" where the transmission rate is increased linearly instead of doubling. [4, 12]

For satellite links, initially, transmission alternates between sending a few segments (TCP's unit of transmission) and stalling, while it waits for the ACKs for these segments to arrive. The naive expectation is that the number of segments sent between each stall would follow a simple exponential pattern: 1,2,4,8,16 etc. However, in practice the actual number of packets transmitted is much smaller, as shown in Table 1 [2].

Stall	Original, Naive Expectation		No delayed ACKs		Delayed ACKs	
	Segments sent	Cumulative segments sent	Segments sent	Cumulative segments sent	Segments sent	Cumulative segments sent
1	2	2	2	2	2	2
2	4	6	3	5	3	5
3	8	14	3	8	5	10
4	16	30	6	14	8	18
5	32	62	9	23	12	30
6	64	126	12	35	18	48
7	128	254	18	53	27	75
8	256	510	27	80	41	116
9	512	1022	42	122	62	178
10	1024	1024	63	185	93	271

Table 1 – Segment sending rate buildup during Slow Start (source: [3])

(Note: The highlighted columns represent the current, common practice as set by the Host Requirements RFC 1123, recommending that all hosts should implement delayed acknowledgements (source [18]).

As can be seen, the unit of time between rate changes is one round trip time (RTT). Seven to 10 RTT's are typically required before switching from Slow Start to Congestion avoidance mode, and then an additional 20-30 RTT's may be required to get back to near link-capacity rate again.

This process leaves a significant amount of bandwidth simply unused. Figure 1 below illustrates this point. As can be seen for a 128 Kbps satellite link, after 16 RTT's (8 Seconds!) sending rate is still only about 80 Kbps. (Assuming ~1 Kbps/segment)

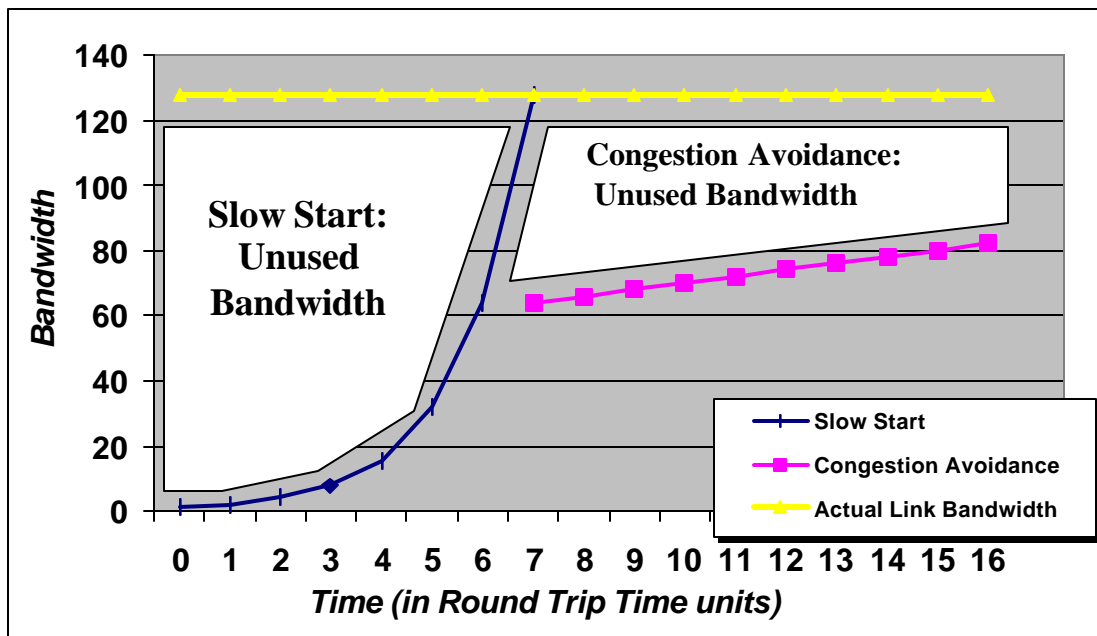


Figure 1 - Effects of Slow Start and Congestion Avoidance

A crucial consideration in this context is the size of the data object(s) transferred during each connection. Frequently, the whole transfer is over long before TCP reaches “cruising speed”. For example, it is estimated that average Web page objects are less than 25 Kbytes in size. Such objects would be retrieved in less than 30 segments, and thus the whole transfer will be over after about six RTT’s and the transmission rate will never exceed about 64 Kbps, regardless of link capacity.

Since standard TCP does not “remember” link capacity between connection, the whole process is repeated each time.

Possible solutions, such as using parallel and persistent connections are discussed below.

Packet Loss Detection and Recovery

The discussion above ignored the effects of packet loss. Packets may be lost due to exceeding link capacity, exceeding buffering capacity in the network, or Bit Errors (BER) on the transmission links. TCP provides a reliable transport service, but examining the details shows that over satellite links it can involve significant speed penalties (as well as bandwidth inefficiencies discussed separately below)

TCP, having no way to detect link Errors (BER) assumes losses due to BER are negligible, and that all packet loss is due to congestion (exceeding link or buffering capacity). When a loss is detected, TCP takes steps to (1) recover the lost data and (2) reduce the assumed congestion level, by reducing dramatically its transmission rate.

Loss detection

TCP detects lost packets in one of two ways - “Time Out” (TO), and “Fast Recovery/Fast Retransmit” (FR/FR) . The “time out” mechanism is simple – when data is sent, an ACK is expected from the receiver within a set time. When this time is exceeded, the packet may be assumed lost.

Since waiting for a time out can mean long periods without activity, an additional mechanism called “Fast recovery/Fast Retransmit” is almost universally used. [12]. if a packet (or more) is lost, but subsequent packets arrive, the receiver generates an ACK packet for each segment seen. Since TCP’s ACK specifies the last correctly received segment, these ACKs will all be duplicates. When the sender receives (at least) three duplicate ACKs, TCP assumes a packet was lost. Note that this mechanism provides a faster response to the loss than waiting for a time-out, but is limited to handling one loss per transmission window.

Due to TCP’s ACK mechanism, losses are discovered one at a time. So, if several packets are lost concurrently, the above mechanisms will correctly detect the first loss, but additional losses will only be detected when the first loss is successfully corrected.

Loss Handling

When a loss is detected, TCP will retransmit the lost segment. In addition, since all loss is assumed to be the result of congestion, TCP will try to reduce the apparent congestion level by reducing its transmission rate. The exact action taken depends on the way the loss was detected. If the loss was detected by a time-out, TCP will enter “slow start” mode, as discussed above. If the loss was detected by the duplicate ACK mechanism, TCP will reduce its sending rate to a little more than 50% of the current rate, and go into “congestion avoidance” mode as discussed above.

Real life experiences show that a majority of packet loss is detected by the “time out” mechanism [3].

An especially painstaking situation occurs when a packet is lost in the early stages of a Slow Start phase, and discovered using the FR/FR mechanism. After re-sending the lost segment, TCP will enter Congestion Avoidance mode, where the Sending rate is only increased by approximately 1 segment every RTT. Under these circumstances, Throughput is so low, and increasing so gradually that users frequently prefer to stop the transfer (throwing away all accumulated data) and restart it, hoping the exponential acceleration of a new Slow Start will provide better service.

“No BER” Assumption

Despite the widespread use of Forward Error Correction (FEC) techniques on satellite links, BER related packet loss is still present, at times significantly [9]. TCP will inappropriately respond to the assumed congestion, dramatically reducing throughput.

Advertised Receive Window

In an attempt to prevent packet loss, it is desirable to make sure the receiver has enough buffer space to store all incoming data. TCP implements a mechanism called “Advertised Receive Window” for this purpose. Under this scheme, the sender can only send a maximum number of bytes as explicitly allowed by the receiver. This scheme is implemented by the receiver including such an allowance (called “Advertised Receive Window” or RWIN) in each packet transmitted to the sender, such as the ACK packets, for example. Thus, a sender gets an allowance, sends the allowed number of bytes, and will then cease sending until the next ACK comes back, carrying the new allowance. From this, it easily follows that the maximum transmission rate possible under this scheme (and ignoring all other factors) is governed by the formula: [12]

$$(1) \quad \text{Max BPS} = (\text{Advertised receive Window}) / (\text{Round Trip Time})$$

For satellite links where RTT is typically 500-700 ms, and given that the default RWIN is 8-16 Kbyte, the maximum possible transmission speed would be 90-256 Kbps. Note that this result is independent of actual link capacity. In other words, this is the top speed possible even for E1 or better links.

It is usually possible to change the value of RWIN (see below). However, standard TCP only allows a maximum of 64 Kbytes, which will still impose a top limit of about 1 Mbps (assuming 500ms delay, and less if delay is larger).

Path Asymmetry

Satellite networks are frequently provisioned in an asymmetric manner, either in terms of allocated bandwidth, or even using a different technology for each direction (e.g. a hybrid satellite/dial-up network). This is especially inherent in Internet/Intranet environments, where traffic patterns typically exhibit intrinsic asymmetry.

Unfortunately, asymmetry has been shown to be a problem for TCP [11].

TCP's sending rate is constrained by the rate at which ACK's arrive from the receiver. A ratio of 47:1 is about the maximum amount of asymmetry that will allow full speed transfers. Given a dial-up link of 33.6 Kbps we will exceed the limit if we want more than about a T1 rate in the forward direction.

As satellite networks adopt DVB technology and reach multi Mbps speeds, the issues related to asymmetry are likely to get more pronounced.

Resource Consumption issues

So far, we have presented various throughput-limiting issues. Resource consumption issues merit a separate discussion. Most existing satellite networks (and their pricing models) were engineered for a short-transaction model. Internet/Intranet via satellite presents a model where each transaction can involve traffic an order of magnitude greater. The prospect of dramatically higher costs (especially coupled with the expected questionable throughput) puts the whole concept of Internet over satellite into doubt.

ACK Frequency

One of the most significant causes for resource waste is the sheer volume of ACK packets generated by TCP. In TDMA based satellite networks, Carrying the ACKs are a very big drain on network resources. Table 2, adapted from [16], illustrates a situation where an Internet application is used at peak times by 300 concurrent users. Assuming each user downloads about two pages every 2 minutes, containing a total of 20 objects of 5 Kbytes average size. As can be seen, about 70% of the total resources are required to support traffic in the “inbound” direction, with ACK traffic consuming about 47% of total resources.

Description	Unit	Inbound	Outbound	System
# of VSAT terminals active during peak period				300
Transactions every S seconds per terminal	Seconds			120
packets/Transaction		50	69	
Bytes/Transaction	Bytes	87	1212	
Net Traffic (excluding Overhead)	bps	86,880	1,672,560	
Nominal Carrier bit rate	bps	64000	2000000	
B/W per carrier	Khz	100	400	
# of Carriers Needed		9	1	
Total BW Needed	Khz	900	400	1,300

Table 2 - TDMA network sizing for Internet/Intranet application
(Highlighted cells denote traffic assumptions)

Variable RTT

TDMA based satellite network present another source of inefficient bandwidth consumption. In these networks, the apparent RTT has a very large variation (due to channel contention – “collisions” and the time it takes to resolve them). Since the RTT estimate is used by TCP as a basis for various time-out calculations, large variations in RTT cause it to incorrectly assume packets were lost, when in fact they were correctly received. In a test done on the Flash Network Satellite link on a lightly loaded VSAT network, this effect caused about 13% of the available bandwidth to be wasted on needless re-transmissions.

Too Many Concurrent Connections

A significant finding for Internet/Intranet via satellite for a large user population (e.g. a large Intranet network, or an ISP) is presented in [8]. Briefly stated, when there are more concurrent connections (flows) in the pipe than the bandwidth-delay product, TCP forces a packet loss rate approaching 50%, while the link utilization stays high and response time to the users suffers large variations. Figure 2 shows the performance and bandwidth consumption of a typical user on a link lightly loaded/heavily oversubscribed link. Figure 3 shows the expected loss rates as a function of the number of concurrent users on a link.

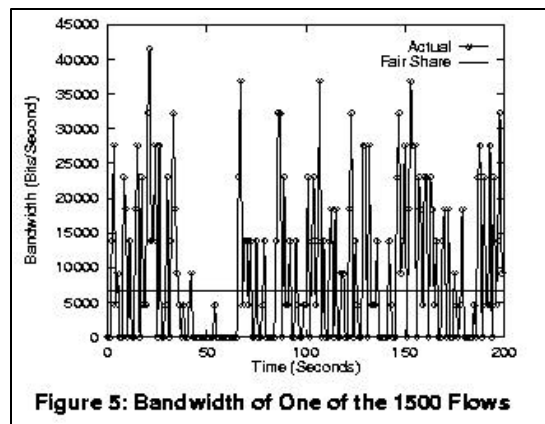
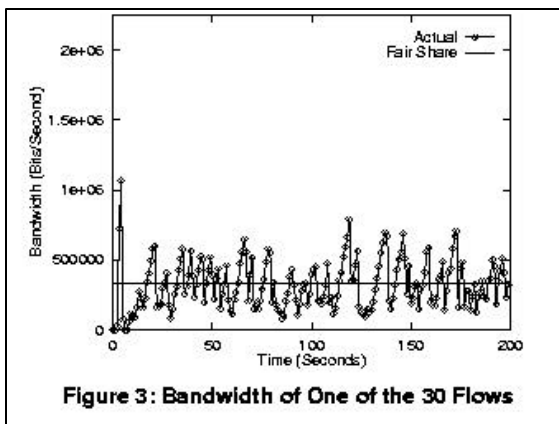


Figure 2 - Performance for one of N users of a link (from [8])

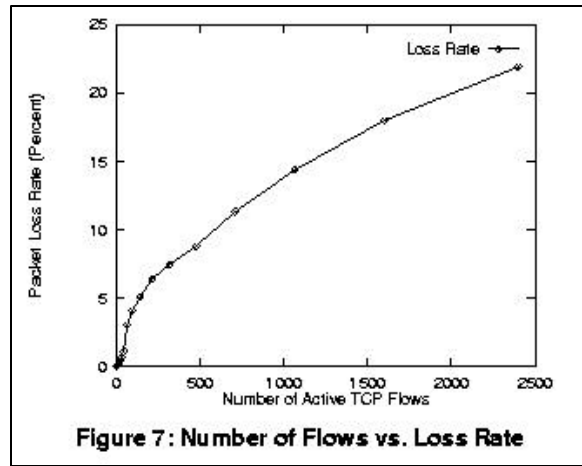


Figure 3 - Loss vs. number of concurrent connections (From [8])

These packet loss rates are a source of both throughput degradation and resource waste. While adding buffering can delay the effect, a complete solution requires the use of a protocol conversion system as described in section below.

HTTP over Satellite: Performance Issues and Solutions

One of the major protocols that need to be considered when planning Internet/Intranet via satellite is HTTP, which is the basis for Browser based applications. As it turns out HTTP has several side effects that have implications for network designers.

HTTP/1.0 uses a separate TCP connection for each object to be retrieved. This causes throughput degradation, mainly due to performing slow-start at frequent intervals and typically terminating the connection long before “cruising speed” is reached. The 5 packets exchanged in connection setup and termination are essentially overhead, and the more connections used, the more bandwidth is wasted. On DAMA based satellite networks careful design is needed to avoid setting up a circuit and tearing it down repeatedly.

Persistent HTTP (P-HTTP) and HTTP/1.1 allow TCP connections to exist for longer periods, and to be used for multiple object retrievals. This cuts down on the connection setup and discard overhead, but on DAMA based satellite network can mean circuits are idle for a significant percentage of the time.

Finally, HTTP (and sometimes the user applications) can force TCP to send segments that are smaller than the maximum allowed size, causing various inefficiencies. [2, 7].

Recent research has suggested a model to estimate the performance of HTTP over TCP [2]. While presenting the full results are beyond the scope of this paper, a simplified first-order approximation is as follows:

$$(2) \text{ Throughput} = (\text{Reply size}) / (\text{Bandwidth} * \text{Delay})$$

Clearly, for satellite networks where Bandwidth * Delay values are inherently large, throughput is likely to be low.

While the choice of HTTP version to be used has to be a mutual choice of both server and client, it is clear that using HTTP/1.1 or at least P-HTTP will provide significantly better performance over satellite links. (But see [5] for a discussion of User-perceived as opposed to actual performance). It is also very beneficial to use a version of HTTP that supports “pipelining” (sending requests before getting the responses to previously sent requests).

If traffic is predominantly HTTP based, disabling Nagle’s algorithm in the TCP stack should be considered. While this may generate a slightly larger number of packets, throughput will usually be better.

While using multiple parallel connections is better than a single connection, current research indicates that the browser default of four is optimal, and using more connections will not provide any significant benefits [6]

TCP over Satellite Solutions

Traffic reduction (caching & compression)

A major improvement is to be gained by simply making sure that TCP has a smaller amount of data that needs to be transported over satellite links.

Compression

A compression system directly reduces the amount of traffic that needs to be transported, thereby saving bandwidth, speeding up response time and shortening transfer times. Compression gains depend on the content transferred. Typical bandwidth savings for general Internet/Intranet browsing are around 30%. For text based data (e.g. E-mail messages) 60% gains are common.

Caching

Consider a corporate Intranet over a satellite link. It is very likely that when a query is carried out, it is a repeat of a previous query. With a caching system, the results of the previous instance of the query are saved, and if they are still up to date, the locally installed cache server will respond. This avoids the need to carry the data over the satellite link again (and again, and again...)

Cache servers are available from many commercial sources and commonly achieve a saving of 30% to 50% of the required bandwidth.

TCP Extensions and Tuning

There are several TCP extensions in various stages of research and implementation that are aimed at solving and mitigating TCP's issues when used over satellite links. The two major TCP extensions to be discussed are RFC 1323 [14] and RFC 2018 [15].

RFC1323 (sometimes called "TCP Large Windows") removes the 'Advertised Receive Window' maximum throughput limitation. RFC 2018 (usually known as 'SACK TCP') allows TCP to handle more than a single packet loss in a transmission window.

The performance of SACK is analyzed by many researches [10, 17]. While there is common agreement that TCP SACK is beneficial, there does not seem to be agreement as to the amount of the gain and the conditions affecting the gain.

Users of an Internet/Intranet via satellite application should use TCP with these extensions, now commonly available (e.g. on Windows 98, Linux) and make sure they are enabled.

Additional TCP extensions are described and discussed in [13] and in an upcoming IETF Internet draft (<http://roland.grc.nasa.gov/~mallman/papers/draft-ietf-tcpsat-res-issues-07.txt>) but are beyond the scope of our discussion.

In addition, various parameters of TCP may be tuned to better match TCP to the nature of the satellite link. While detailing, these tunings are beyond the scope of our discussion, a good starting point is http://www.psc.edu/networking/perf_tune.html.

Using a NON-TCP Protocol over the Satellite Link.

A major consideration guiding TCP extensions proposed through the IETF is that the extensions must be safe and beneficial (or at least non-harmful) in all cases. When a TCP extension is proposed, it must stand the test of "what if everyone in the shared, public network adopted this extension?" This consideration, forces all TCP extensions to maintain the slow-start and congestion avoidance mechanisms (perhaps with slight modifications) to prevent a congestive collapse of the network, and limits the range of the improvements that can be realized.

If we limit our interest to a closed, private network domain, than many TCP issues may be solved by essentially replacing TCP over the satellite link with a different protocol.

TCP Spoofing

TCP spoofing is a simple form of protocol replacement. A “spoofing” system involves a device at the “near end” of the Satellite link pretending to be the intended destination for the TCP connection, returning ACKs, etc. Data sent to that device is transferred over the satellite link to the “far end”, where a device playing the role of the original source builds a separate TCP connection to the intended destination. The data is sent over the space segments using a private protocol, typically the inherent data-link protocol in use by the satellite access equipment. Since reliability is typically provided at the data-link layer, TCP is modified, or even completely eliminated to get acceptable throughput and bandwidth consumption.

“Spoofing” systems are typically available as an integral part (or at least as an option) of the satellite equipment. The performance and savings provided vary greatly, and can be hard to determine (certainly, equipment makers are not easily convinced to provide this information).

It is therefore necessary to carefully test these systems for suitability in conditions as close to the intended environment and target application as possible.

Full Protocol Conversion

Protocol conversion systems are a form of “super spoofing” that provide all the benefits of spoofing described above, including throughput gains and resource savings, plus additional benefits. For example SatBooster, a Protocol conversion system made by Flash Networks provides dramatic throughput gains while using an order of magnitude less ACKs. In addition, SatBooster provides compression, rate control and user-mandated bandwidth allocation, and load balancing fault-tolerant configurations. Figure 4 shows the results of a recent test comparing the throughput of a HTTP based application using TCP and SatBooster over a satellite link, excluding compression gains. As can be seen SatBooster can provide 100% to 500% speed gains, while also being more tolerant of packet loss and link BER.

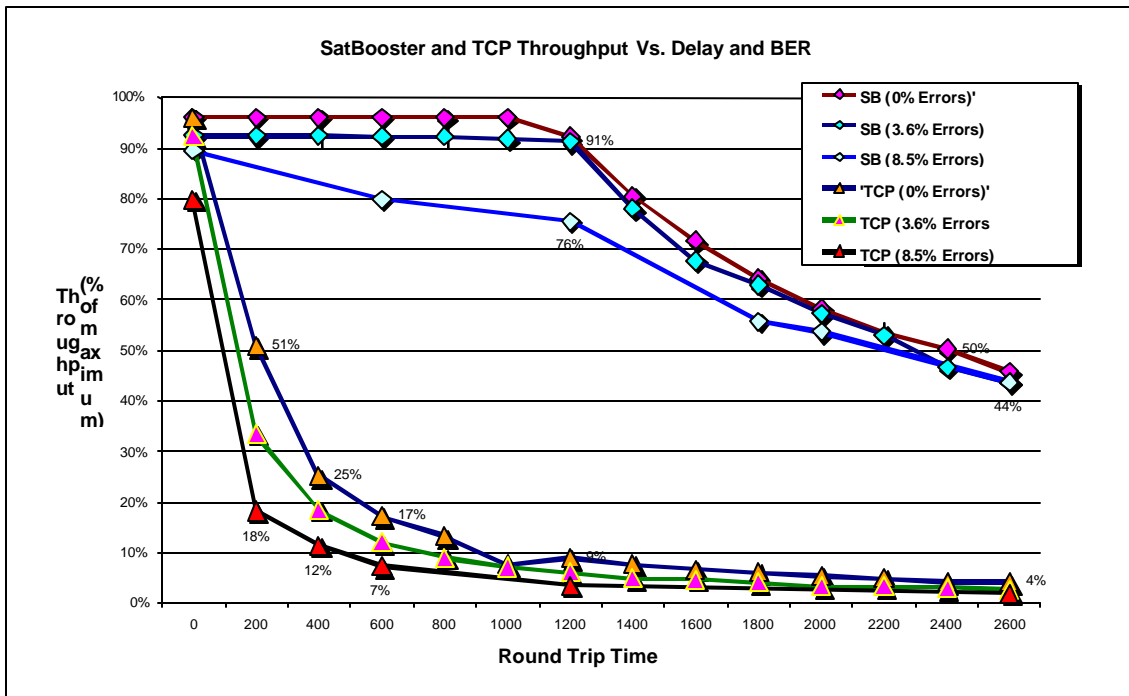


Figure 4 - TCP and SatBooster performance as a function of Delay and BER

Cost Justification

While these proposed solutions can be easily warranted by quantifying the effects of the time savings and improved human resource productivity resulting from the higher transmission speed, ideally, it should be justified with direct cost savings. In some cases, especially TDMA or DAMA based networks, this cost savings can be shown to be the justification.

Consider, for example, the effects of using SatBooster in the same scenario as described in table 1. The fact that SatBooster typically only needs to send about 10% of the acknowledgements, as compared to TCP (while keeping full reliability), has an immediate effect on the bandwidth requirements, and hence on the monthly communication bill.

Description	Unit	Inbound	Outbound	System
# of VSAT terminals active during peak period				300
Transactions every S seconds per terminal	Seconds			120
packets/Transaction		50	69	
Bytes/Transaction	Bytes	87	1212	
Net Traffic (excluding Overhead)	bps	86,880	1,672,560	
Nominal Carrier bit rate	bps	64000	2000000	
B/W per carrier	Khz	100	400	
# of Carriers Needed		9	1	
Total BW Needed	Khz	900	400	1,300

Table 3 - TDMA network sizing for Internet/Intranet application

(Note: The highlighted cells denote the assumption that each user downloads about two pages every 2 minutes, containing a total of 20 objects of 5 Kbytes average size. The numbers are derived from these assumptions and are based on TCP's behaviour.)

This savings is clearly seen in table 2, below, where the same traffic model is used as in table 1, except that the packet counts and average sizes are as would typically be seen using SatBooster. As can be seen by comparing tables 1 and 2, bandwidth requirements are reduced from 1.3 MHz to 0.7 MHz and the bandwidth consumed by ACKs is reduced from 47% to 23%. (It is interesting to note that using SatBooster the 1.3 MHz bandwidth consumption from Table 1 would be almost exactly enough to support an additional 100 users at peak time).

Description	Unit	Inbound	Outbound	System
# of VSAT terminals active during peak period				300
Transactions every S seconds per terminal	Seconds			120
packets/Transaction		9	69	
Bytes/Transaction	Bytes	191	1212	
Net Traffic (excluding Overhead)	bps	34,400	1,672,560	
Nominal Carrier bit rate	bps	64000	2000000	
B/W per carrier	Khz	100	400	
# of Carriers Needed		3	1	4
Total BW Needed	Khz	300	400	700

Table 4 - Effect of using SatBooster on TDMA network sizing from table 1

In Conclusion

TCP and HTTP are the defacto protocols widely utilized in today's data communication. In certain environments, such as satellite communication however, TCP's inherent characteristics, and some of HTTP's attributes do not utilize the bandwidth efficiently and do not provide for high throughput. Fortunately solutions exist to combat the shortcoming of TCP over satellite links. With some careful deliberation, finding the right solution means that Internet via satellite can be effective and efficient.

References

- [1] Enhancing TCP Over satellite Channels using Standard Mechanisms, Request For comments, RFC 2488
M. Allman, D. Glover and L. Sanchez,
Internet Engineering Task Force (IETF), 1999
<http://www.ietf.org/rfc/rfc2488.txt>
- [2] Modeling the performance of HTTP over several transport protocols
John Heidemann, Katia Obraczka, and Joe Touch;
IEEE/ACM Trans. Networking 5, 5 (Oct. 1997), Pages 616 – 630
<http://www.acm.org/pubs/citations/journals/ton/1997-5-5/p616-heidemann/>
- [3] Modeling TCP throughput: a simple model and its empirical validation
Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose;
Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication , 1998, Pages 303 – 314
<http://www.acm.org/pubs/citations/proceedings/comm/285237/p303-padhye/>
- [4] The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm
M. Mathis, J. Semke, J. Mahdavi, T. Ott,
Computer Communication Review, volume 27, number 3, pp. 67-82, July 1997.
http://www.psc.edu/networking/papers/model_ccr97.ps
- [5] Network and User-Perceived Performance of Web Page Retrievals
Hans Kruse, Mark Allman, Paul Mallasch.
November, 1998. Proceedings of the First International Conference on Telecommunications and Electronic Commerce (ICTEC).
<http://roland.grc.nasa.gov/~mallman/papers/ecom98.ps>
- [6] HTTP Page Transfer Rates Over Geo-Stationary Satellite Links.
Hans Kruse, Mark Allman, Jim Griner, Diepchi Tran.
Proceedings of the Sixth International Conference on Telecommunication Systems, March 1998.
<http://roland.grc.nasa.gov/~mallman/papers/nash98.ps>
- [7] Network Performance Effects of HTTP/1.1, CSS1, and PNG
H. F. Nielsen, J. Gettys, A. Baird-Smith, E. Prud'hommeaux, H. Wium Lie, C. Lilley
World Wide Web Consortium (W3C), June 1997
<http://www.w3.org/Protocols/HTTP/Performance/Pipeline.html>
- [8] TCP Behavior with Many Flows,
Morris, R.,
IEEE International Conference on Network Protocols, October 1997, Atlanta, Georgia
<http://www.eecs.harvard.edu/networking/papers/icnp97-web.ps>
- [9] HTTP Experiments in Satellite-Based Networks.
M. Allman,
Presented at the Telecommunications Industry Association Meeting. March 1999.
<http://roland.grc.nasa.gov/~mallman/papers/tia-3.99.ps>
- [10] TCP Performance Over Satellite Links.
Mark Allman, Chris Hayes, Hans Kruse, Shawn Ostermann.
Proceedings of the Fifth International Conference on Telecommunications Systems, Nashville, TN, March, 1997.
<http://roland.grc.nasa.gov/~mallman/papers/nash97.ps>

- [11] The effects of asymmetry on TCP performance
Hari Balakrishnan, Venkata N.Padmanabhan, and Randy H.Katz
Proceedings of the third annual ACM/IEEE international conference on Mobile computing and networking , 1997, Pages 77 – 89
<http://www.acm.org/pubs/citations/proceedings/comm/262116/p77-balakrishnan/>
- [12] TCP/IP Illustrated, Volume 1
R.W. Stevens
Addison-Wesley, 1993
- [13] TCP/IP Illustrated, Volume 3
R.W. Stevens
Addison-Wesley, 1996
- [14] TCP extensions for high Performance, Request for comments, RFC 1323
V. Jacobson, R. Braden, D. Borman
Internet Engineering Task Force (IETF), may 1992
<http://www.ietf.org/rfc/rfc1323.txt>
- [15] TCP Selective Acknowledgement Options, Request for comments, RFC 2018
M. Mathis, J. Mahdavi, S. Floyd and A. Romanow
Internet Engineering Task Force (IETF) October 1996
<http://www.ietf.org/rfc/rfc2018.txt>
- [16] The book on VSATs 2, Appendix A
Gilat satellite Networks, 1995
- [17] Simulations based comparisons of Tahoe, Reno and SACK TCP
Kevin fall and Sally Floyd
IEEE Computer Communications Review, July 1996
<http://www.acm.org/sigcomm/ccr/archive/1996/jul96/ccr-9607-fall.html>
- [18] Requirements of Internet Hosts - Communication Layers, Request for comments, RFC 1122
R.T. Broden
Internet Engineering Task Force (IETF), Oct 1989
www.rfc-editor.org/cgi-bin/rfcsearch.pl

About the Author

Michael Orr joined *Flash Networks* with fourteen years of experience in R&D and Project and Product Management. His past experience includes military R&D project management, compiler writing and UNIX system programming. He has worked with embedded real-time software design and development, as well as product management and marketing for a LAN equipment manufacturer. Michael has also consulted and managed technical support in the United States for an expert-system maker. In 1984, Michael graduated with a Bachelor of Science from The Technion, Israel Institute of Technology. Currently, he is involved in R&D and technology directives for *Flash Networks Ltd.* He is married and the proud father of a two year old future hacker.

About the BoosterWare™ Product Family

Flash Networks' proprietary BoosterWare technology accelerates the performance of TCP/IP applications and services in Internet and intranet environments by maximizing throughput, while increasing the reliability of the Internet connection. The BoosterWare product suite includes SatBooster™ and BackBone Booster™ software. SatBooster increases Internet transmission speeds over satellite connections by up to five-fold, whether browsing or downloading files. SatBooster dramatically enhances the performance and reliability of VSAT (very small aperture terminal) installations for satellite communications. BackBone Booster is targeted toward Internet service providers, corporate intranet and virtual private network applications where data transmissions are routed over long backhaul links. BackBone Booster increases Internet transmissions by up to five-fold by translating TCP/IP connections into BoosterWare packets over the backhaul portion of the network, and then retranslating back to TCP/IP.

About Flash Networks

Flash Networks is a pioneering developer of Internet performance enhancement software that enables satellite carriers, Internet service providers and corporate intranet users to more effectively utilize existing bandwidth for high speed TCP/IP connections. Founded in 1996, the company has customer presence in North America, Europe and Israel and maintains offices in the Herzliya, Israel and Washington, DC high technology corridors. Further information can be found at www.flash-networks.com