

Effects of Networking Configurations on Space Channel Throughput

Stephen Horan and Ruhai Wang
 Manuel Lujan Space Tele-Engineering Program,
 Klipsch School of Electrical and Computer Engineering,
 New Mexico State University, Las Cruces, NM 88003
 (505) 646-3012, shoran@nmsu.edu, ruhwang@nmsu.edu

Abstract. The throughput results for file transfers using file sizes ranging from 1 Kbytes through 1 Mbytes using both the standard TCP/IP and SCPS protocol stacks are reported here. The file transfers are performed at a range of balanced and unbalanced link speeds and channel error rates. The host computers used in the simulations are using standard Linux operating system configurations on PC platforms. The PPP is used as a framing protocol to support communications between the computers. The file-transfer throughput results will show the effects of link configuration and channel error rate on the necessary time to transfer a file. Because the operating system can be configured with different options for the protocols, for example enabling packet header compression, these options need to be factored into the comparison. As part of the throughput reporting, we will show the effects of header compression and selection of congestion algorithm upon the results. The TCP/IP ftp and SCPS-FP using VJ congestion control algorithm results give similar results and better results than SCPS-FP with the Vegas congestion control algorithm in these experiments. No noticeable delay effects were noted with links delays corresponding to GEO orbits with file transfers of 1 Mbytes.

Introduction

The use of Internet-type protocols for space communications is no longer considered a “new” topic of investigation. Research on the subject has been conducted at NASA, DoD, contractor, and university facilities for several years now. At New Mexico State University (NMSU), we have been concentrating on the performance of two protocols suites: the file transfer protocol (ftp) that is part of the Transmission Control Protocol/Internet Protocol (TCP/IP) stack and the file protocol (fp) that is part of the Space Communications Protocol Standard (SCPS) [1] developed under the Consultative Committee for Space Data Systems (CCSDS) standards process. In the NMSU studies, we have been concentrating

on the performance of the protocols in a small satellite environment. By that, we mean that the anticipated user would be interested in the following characteristics and capabilities:

1. Forward and return data rates from 2400 bps through 100 kbps, selectable in each direction,
2. Bit Error Rate (BER) selection from 0 through 0.0001, selectable in each direction,
3. Data file sizes from 1 Kbytes through 1 Mbytes,
4. Short transfer opportunities, e.g., typical 5-minute ground station passes,
5. User-selectable delay times for the forward and return link up to 5 seconds.

We have developed a Space-to-Ground Link Simulator (SGLS) to provide the simulation capabilities to test both protocol suites ([2], [3], and [4]). During the testing, we discovered that the protocol suites have a number of options that affect the overall protocol performance more greatly than others. This report discusses the experiments performed to map out how the options affect the overall performance from the user's point of view. When possible, option performance on both suites of protocols will be compared in a fair a manner as possible.

This investigation is intended to look at the performance of the TCP/IP and SCPS protocol suites in a simulated space channel environment. In particular, we are interested in investigating the similarities and differences caused by congestion control algorithms, compression options, and time stamp options on the protocol performance. We will be running these investigations with channel BER, link delay, and file size as variables that can be controlled to simulate various user needs.

The range of investigations is limited to the following parameter space:

1. Data file transfers from 1 Kbytes through 1 Mbytes
2. Return link speeds of 115200 bps and forward links speeds of 115200 bps for symmetric links and 2400 bps for unsymmetric links
3. PPP as the data link layer framing protocol
4. Channel bit error rates of 0, 10^{-6} and 10^{-5} due to Additive White Gaussian noise on the channel
5. Channel delays of 0, 3 ms, 120 ms, and 1280 ms

6. $\text{f} \text{t} \text{p}$ header compression enabled or disabled
7. $\text{f} \text{p}$ time stamps enabled or disabled
8. $\text{f} \text{p}$ VJ or Vegas congestion control algorithm
9. Single-hop point-to-point data links

In particular, these results may not be applicable to networks with significantly different bandwidth-delay products. The simulator models links that are composed of a single ground station and a single satellite. There are no relay satellites or external control centers modeled in this configuration. Because we have an emphasis on short-term file transfers, the interactions that may occur in long file transfer processes are not covered here.

Methods and Procedures

The SGLS channel simulator is used to perform the error generation and link delay used to test the protocol suite performance. In this section, we describe the simulator and the standard tools for gathering data to measure the performance.

The Space-to-Ground Link Simulator (SGLS) has been developed at NMSU to model space channel characteristics experienced in transmitting data. The simulator is described fully in [2], [3], and [4]. Basically, the SGLS configuration allows the user to configure the simulated channel to

1. Allow for simultaneous bi-directional data flow (forward and return channels),
2. Allow user-selectable error rates and statistical descriptions of the channel,
3. Allow different data rates on the forward and return links as would be

- found in satellite links, e.g. 2400 baud forward, 115200 baud return, and
4. Provide for a simulated delay up to 5 seconds on each link.

The SGLS utilizes the LabVIEW programming language to control data throughput through the simulator, mix the baseband data stream with the user-selected error vector, and provide for the user-selectable link delay value. The hardware configuration is illustrated in Figure 1. The LabVIEW software is run as an application on each of the SGLS computers. Typically, the LabVIEW modules are the only applications software running on the computers. This configuration was developed to model point-to-point satellite links in its current configuration. The bandwidth-delay product for the system under a 115200 bps symmetric link with no imposed channel delay is 671 bytes. As a comparison, a T-1 line crossing the United States has an estimated bandwidth-delay product of 11,580 bytes [5]. Therefore, this simulator corresponds to a relatively low capacity system.

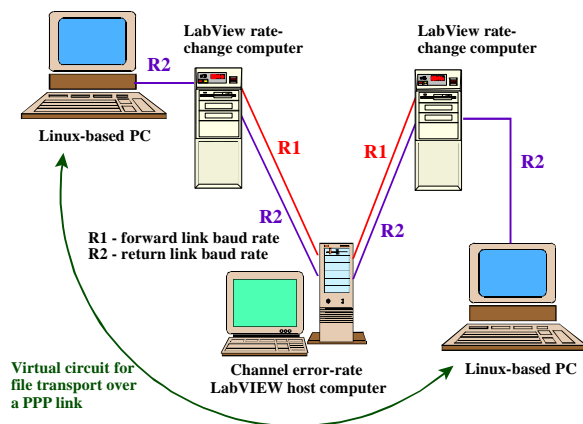


Figure 1 - SGLS Hardware configuration.

The three PCs in the SGLS are Dell 600-MHz computers with 128 Mbytes of memory running Windows 98 second edition. The first

Linux-based PC is a Dell 266 MHz computer. This is our logical ground station computer. The second Linux-based PC is a Gateway 166 MHz. This is our logical satellite computer. Both Linux computers are running Red Hat Linux version 6.1. The SGLS is connected to the Linux computers using serial cables connected to the COM serial ports on each computer. The data connections are configured without hardware or software handshaking to allow for a simulation that would be similar to interfacing with a satellite radio system. In all cases, the links between the SGLS and the Linux computer were set to 115200 bps (R2 in Figure 1). The simulations run with symmetric links had the forward link (R1 in Figure 1) also set to 115200 bps. The simulations run with unsymmetric links had the forward link set to 2400 bps. Other combinations are possible and the reader should refer to [3] for representative results.

Two Expect scripts were modified based on models provided by MITRE [6]. They were developed to automate SCPS-FP and ftp tests in NMSU test-bed. They basically achieve the following objectives:

1. Automate file transfers and gather reported transmission times for multiple runs at different file sizes for both protocols in one experiment configuration;
2. Dump traffic over PPP interface for each connection for both protocols using existing tool tcpdump that is supplied with Red Hat Linux;
3. Conduct tests in (1) with various error rates under human intervention to set the BER in the SGLS;
4. Achieve all the above three objectives over different interfaces (e.g., Ethernet, ATM) after trivial modifications.

The software used in these experiments is used “as is” from the suppliers without any attempt to optimize or modify it. The only changes are to select options as described in the experiments. It is felt that this would most closely resemble the situation used by most system developers who are more concerned with satellite development than attempting to fine tune software. The operating system used on the source and destination data computers is Red Hat Linux version 6.1. The kernel build is 2.2.12-20. The TCP/IP and PPP protocols are those that come with the Red Hat installation software. Both are installed in the kernel without modification. The SCPS protocol suite is based upon the software provided by MITRE [7]. All tests described here were conducted with version 1.1.34 of the SCPS software.

Each experiment configuration was tested using 10 runs through the simulator. It is assumed that this is sufficient to characterize the behavior. Most satellite transfers can be thought of as single-attempt trials. The network would have no memory of previous results or chance to optimize based on previous data streams. It is assumed that 10 back-to-back tests will be representative of these single-shot attempts at data transfers. The test configuration can be thought of as a single point-to-point transmission between a ground station and a satellite. There is no external network interaction. It is assumed that the important parameters for this investigation are contained in this link and not in other ground links.

There are several options directly tested with these experiments. The basic configuration to be examined is to look at TCP/IP `ftp` using the VJ congestion control and the SCPS-FP also using the VJ congestion control. Both protocols will have the time stamp option

enabled. This gives the “fair” baseline comparison with both protocols configured as identically as possible. In these tests, the data throughput will be measured with simulated channel BERs corresponding to 0 errors, 0.000001 and 0.00001. These will be run with the forward and return data links configured for symmetric and unsymmetric transfers. No link delay will be used in these transfer experiments.

The default congestion control option for the SCPS suite is to use the Vegas congestion control while the default for TCP/IP is the VJ control algorithm. The next investigation is to look at how the VJ and Vegas congestion control algorithms available for SCPS react to the presence of channel errors as in the baseline case. These will be compared with the TCP/IP results as well.

The TCP/IP `ftp` protocol can be used with header compression enabled or disabled. Tests will be conducted with both options. This option does not exist for SCPS under the current Linux configuration.

The SCPS `fp` protocol can be used with time stamps either enabled or disabled. The current Linux configuration enables time stamps for `ftp` by default. The SCPS protocol will be tested with the time stamps both enabled and disabled.

The user can set the desired link delay from 0 ms through 5000 ms (5 seconds). For these simulations, the simulated link delays of 0 ms, 3 ms, 120 ms, and 1280 ms were used on each the forward and return links. These link delays correspond to no delay, LEO satellite orbit, GEO satellite orbit, and lunar orbit.

The Expect scripts produce data files that contain the description of the experiment, the

file size being transmitted, delay time, BER setting, and transfer time for each run. These computer-generated data files are then analyzed for data throughput times. The analysis used here is to look at the throughput times based on the 10-run averages for each experiment configuration. The analysis is computed using Excel spreadsheets and the results are plotted using Corel Presentations.

Results and Discussions

The full details of the experiments and theory behind the algorithms are contained in the companion report [8]. The results reported here contain the important points.

Baseline Configuration

The baseline tests include experiments with TCP/IP f_{tp} and SCPS f_p under the BER conditions of 0 errors, 0.000001, and 0.00001, no channel delays, time stamps enabled, and both congestion control algorithms. The f_{tp} protocol uses only the Van Jacobson congestion control algorithm while the f_p protocol can use either the Van Jacobson or the TCP Vegas congestion control algorithms with the TCP Vegas algorithm being the default mode. Figure 2 shows the results for f_{tp} , Figure 3 shows the results with f_p using the VJ congestion control, and Figure 4 and shows the results with f_p using the Vegas congestion control. Each figure shows the results for symmetric (115200 bps on both the forward and return links) and unsymmetric (115200 bps on the forward link and 2400 bps on the return link) data link configurations.

From Figure 2, we can see that the symmetric links generally run faster than the unsymmetric links as would be expected due

to the much higher ACK rate on the symmetric links. For the 100-Kbyte and 1-Mbyte experiments, the unsymmetric links ran approximately 74% slower than the symmetric links. The 1-Kbyte case is the exception here but we know from `tcpdump` results that these file transfers are completed in a single packet. For the 1-Kbyte experiments, we are primarily seeing the variations within the operating system and the computer and not necessarily link effects. While the variations appear to be large, this is plotted on a logarithmic scale and the transfer times are a fraction of a second.

As we had seen in earlier experiments, the file transfer slows down under the higher error conditions, that is the BER is 0.00001, and the throughput curve turns sharply upward for the 100-Kbyte and 1-Mbyte file sizes. This indicates that the errors are frequent enough to cause the link throughput to be reduced by the congestion control algorithm.

By comparing Figures 3 and 4, we see that both congestion control algorithms have about the same throughput for the f_p protocol at each BER and file size. The overall characteristics of the f_p results are similar to that of the f_{tp} experiments for both the symmetric and unsymmetric links. This would be expected for the f_p protocol with the Van Jacobson congestion control algorithm since that algorithm is also used by f_{tp} . By examining the results for the 1-Mbyte experiments, we see that the f_p protocol with the Van Jacobson congestion control algorithm runs within 2% of the results for the f_{tp} protocol. This difference is within the variance of the experimental results. Interestingly, the f_p Vegas congestion control algorithm does not perform significantly better than does the f_{tp} congestion control algorithm in these experiments. In fact, looking at the 1-Mbyte

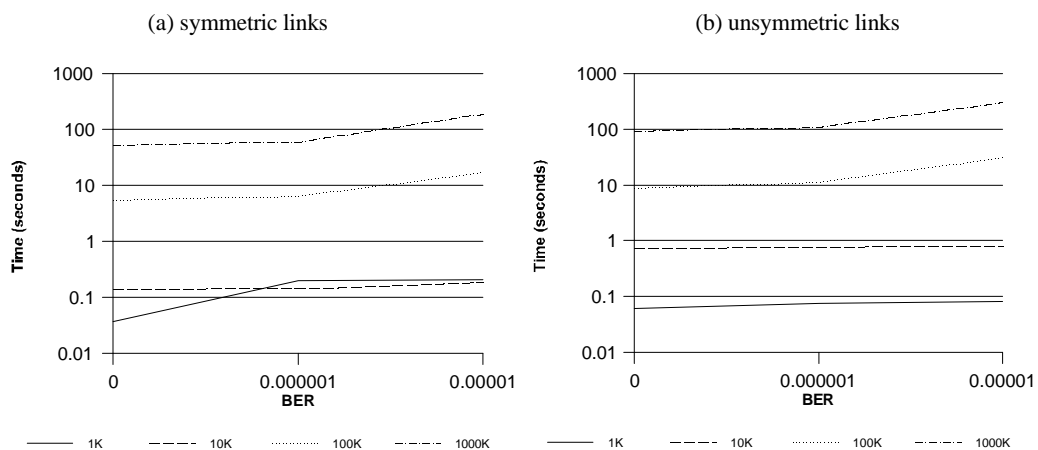


Figure 2 – Baseline ftp experiment results for symmetric and unsymmetric links as a function of file size and link BER.

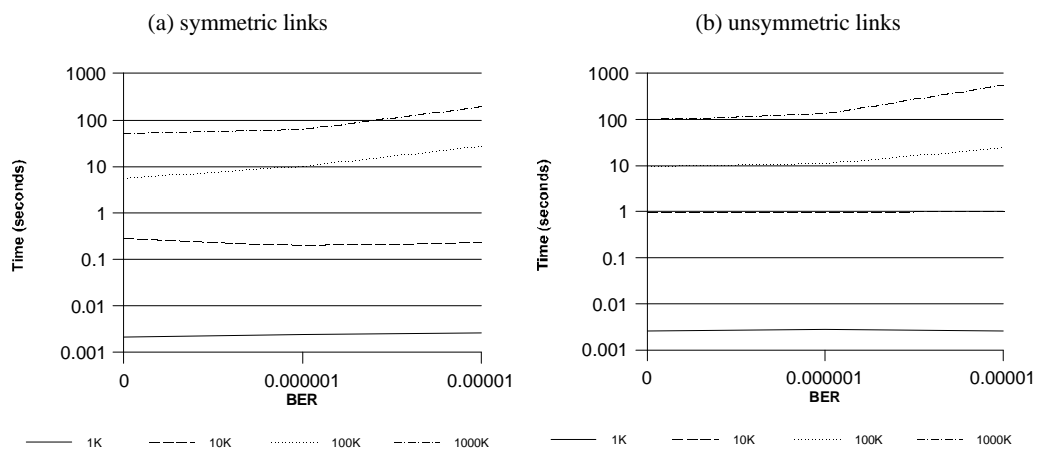


Figure 3 – Baseline fp experiment results for symmetric and unsymmetric links as a function of file size and link BER.

file transfer experiments, using fp with the Vegas control algorithm runs significantly slower than does ftp and at the 10^{-5} BER, the fp protocol is taking twice the time to complete the file transfer as does ftp.

We may conclude that TCP Vegas achieves around half the throughput of VJ at a high BER in our simulation environment. That would be an interesting test to repeat in an actual satellite environment.

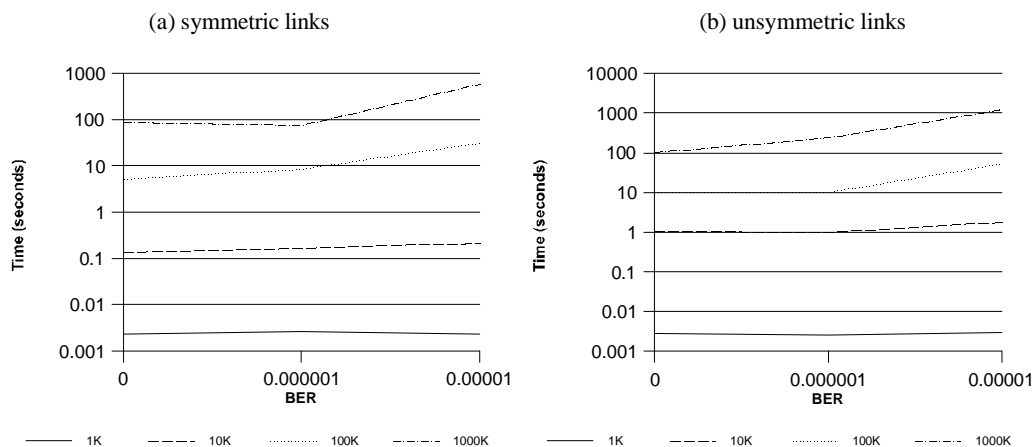


Figure 4 – Baseline f_p experiment results for symmetric and unsymmetric links as a function of file size and link BER and the Vegas congestion control algorithm.

Effect of Time Stamps

As explained in the Introduction, the SCPS-FP protocol can have time stamps enabled or disabled as a protocol option. In this set of experiments, we look at the effects of the time stamps without delay in the link and with delay in the link. In both cases, we examine the effects of the Vegas and Van Jacobson congestion control as well. Because the TCP/IP has no time stamp options available in our testbed, time stamp effects on the TCP/IP protocol will not be examined in this set of experiments.

Effect of Time Stamps Without Link Delay

The first case we examine is the case where there is no link delay. Here we examine the f_p throughput as a function of channel BER and file size. The results will first look at the Van Jacobson congestion control and then the

Vegas congestion control with the SCPS f_p protocol.

VJ Congestion Control

The first case to be examined is the Van Jacobson congestion control. Figure 5 shows the throughput results for symmetric data links and Figure 6 shows the throughput results for unsymmetric links. From Figure 5, we can see a slight improvement in throughput with time stamps disabled over having them enabled for large files (100 Kbytes and 1 Mbytes) and with the larger channel errors (BER = 0.00001). For the 100-Kbyte and 1-Mbyte files, having the time stamps disabled allows the link to run approximately 22% faster than with the time stamps enabled for symmetric data links. For the smaller files sizes, there is no significant difference. Based on tcpdump output plots from other experiments, we interpret this lack of a difference as being due to the files being sent in basically a single window group.

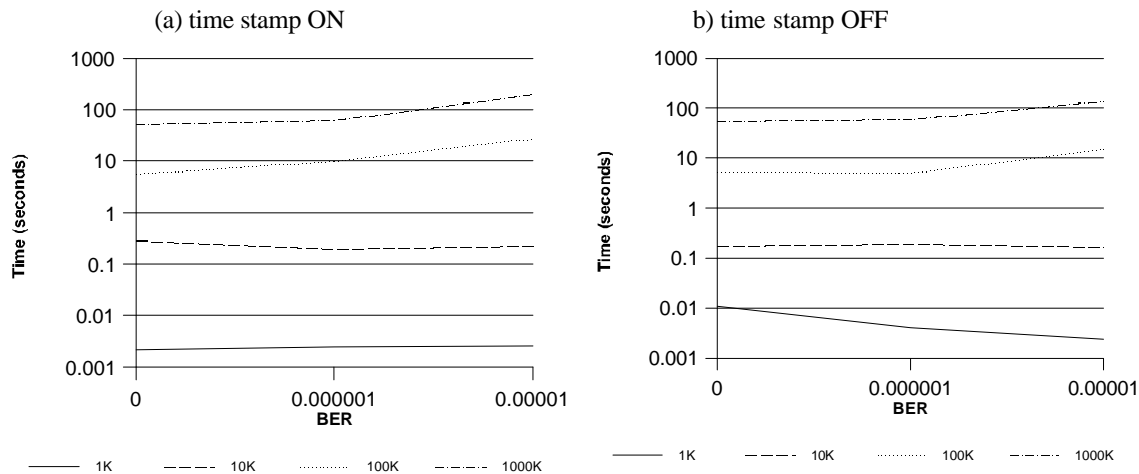


Figure 5 – Time stamp effect on throughput for the f_p protocol using the VJ congestion control on symmetric links as a function of file size and BER.

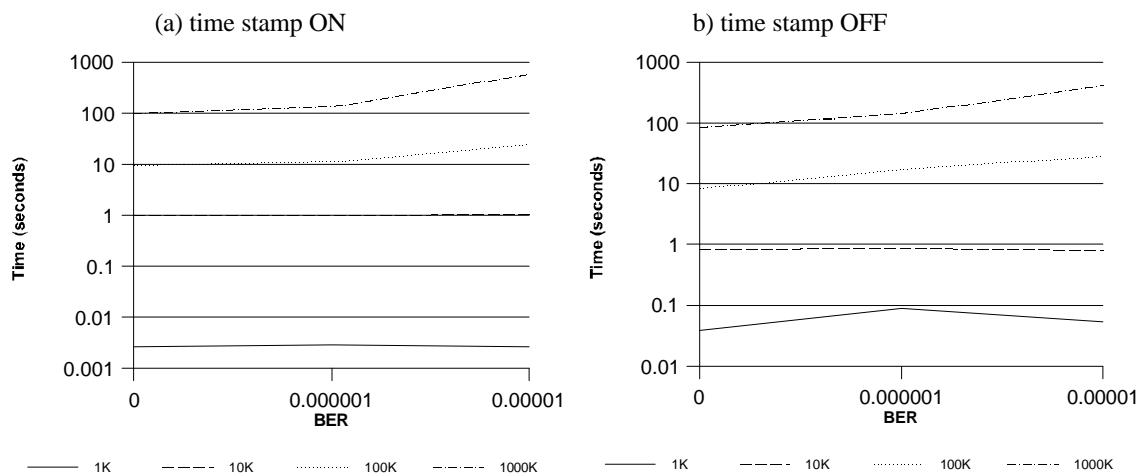


Figure 6 – Time stamp effect on throughput for the f_p protocol using the VJ congestion control on unsymmetric links as a function of file size and link BER.

Therefore, the time stamp does not have a chance to interact with the transmission process.

For the unsymmetric links illustrated in Figure 6, there are mixed results in the throughput with the time stamps disabled or enabled. For the 1-Mbytes file experiments, the 0-error case and the 10^{-5} BER case ran faster with time

stamps disabled while the other cases, the time stamp disabled experiments ran slower. In the no-error case and high-error, we see the advantage to having less header data to send because there are either no re-transmissions or several re-transmissions. With fewer re-transmissions than the high-error case, the slow ACK process dominates the smaller header size.

Vegas Congestion Control

The second case to be examined is the Vegas congestion control. Figure 7 shows the throughput results for symmetric data links and Figure 8 shows the throughput results for unsymmetric links. As was the case with the VJ congestion control, we can see in Figure 7, that there is an improvement in throughput for large files with time stamps disabled and with the larger channel errors on symmetric links. In these experiments, the 100-Kbyte files showed a 20% lower transfer time while the 1-Mbyte files showed a 59% faster time. For the smaller files sizes, there is no significant difference.

For the unsymmetric links illustrated in Figure 8, there is an average of a 20% decrease in the throughput time with the time stamps disabled or over the times with the time stamps enabled. The largest improvement is for the 10^{-5} BER case where the 1-Mbyte file transfer experiments ran 47% faster having the time stamps disabled. Furthermore, we can see that there is no significant difference in the link throughput between the VJ and Vegas congestion control algorithms with time stamps either enabled or disabled.

Generally, the Vegas congestion control algorithm performed worse in these experiments than did the Van Jacobson congestion control algorithm. For the symmetric link experiments, the Vegas algorithm ran 47% slower on the average than the Van Jacobson algorithm for the 100-Kbyte and 1-Mbyte file experiments. For the unsymmetric link experiments, the Vegas algorithm ran 20% slower on the average than the Van Jacobson algorithm for the 100-Kbyte and 1-Mbyte file experiments.

Effect of Time Stamps With Link Delay

To observe the effects of link delay, we conducted a series of experiments corresponding to link delays encountered by LEO, GEO, and lunar satellites. For these experiments, the BER was fixed at 0 link errors. Again, we differentiate between the Van Jacobson and Vegas congestion control algorithms for the `fp` protocol.

VJ Congestion Control

The effects of time stamps with link delay with the Van Jacobson congestion control algorithm is illustrated in Figure 9 for symmetric links and Figure 10 for unsymmetric links. In these figures, we notice several interesting effects. Firstly, the operating system variations can be seen again for the 1-Kbyte files. Secondly, the system treats files larger than 10 Kbytes different than files smaller than 10 Kbytes as can be seen by the change in slope of the curves at this point. We attribute this to the interaction of the packet window algorithm with the transfer process. For smaller file transfers, they can be completed in a single transmission window while the larger files interact with the window algorithm more significantly. Finally, for large data files, there are no significant differences in the throughput times for the 3-millisecond delays and the 120-millisecond delays. This implies that links through GEO orbital distances will not see significant throughput differences over the terrestrial case for these large files over relatively error-free links. This is explained by noting that the acknowledgment process and operating system interactions are more important in determining the throughput than the point-to-point link delay for these large files. For the small files, the link delay is on the order of

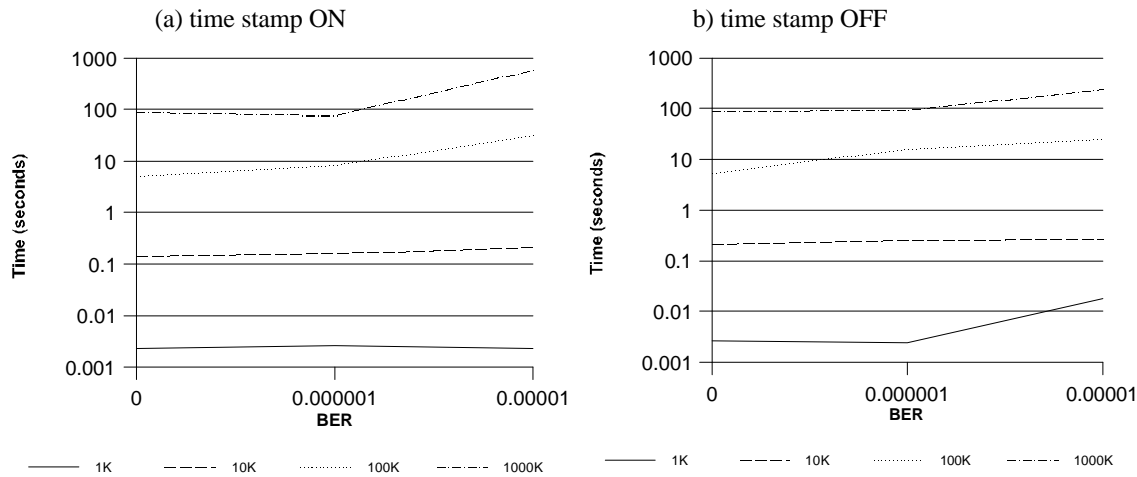


Figure 7 – Time stamp effect on throughput for the fp protocol using the Vegas congestion control on symmetric links as a function of file size and link BER.

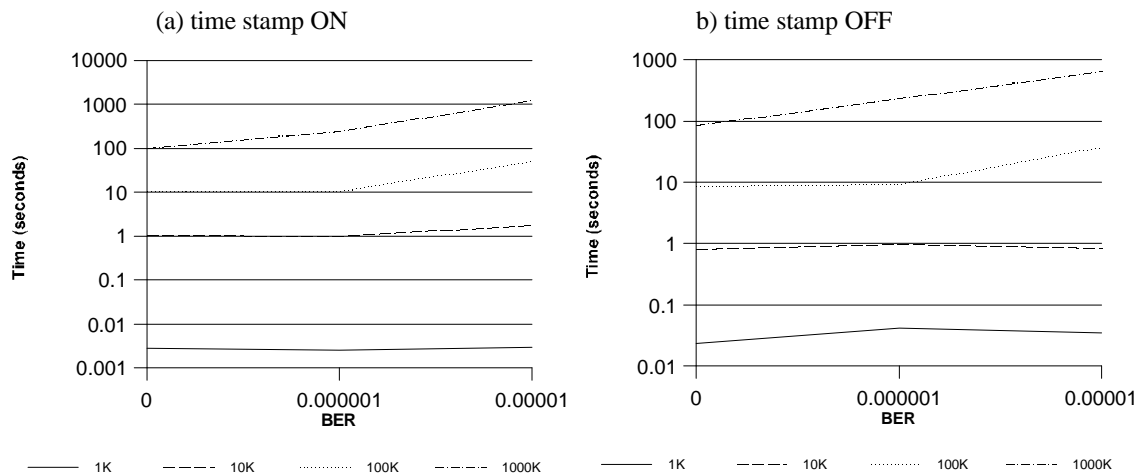


Figure 8 – Time stamp effect on throughput for the fp protocol using the Vegas congestion control on unsymmetric links as a function of file size and link BER.

magnitude of the file transfer time so the link delay is relatively more important.

Vegas Congestion Control

The effects of time stamps with link delay with the Vegas congestion control algorithm is

illustrated in Figure 11 for symmetric links and Figure 12 for unsymmetric links. In these figures, we notice several interesting effects that mirror those found with the Van Jacobson congestion control. Firstly, the operating system variations can be seen again for the 1-Kbyte files. Secondly, the system treats files larger than 10 Kbytes different than files

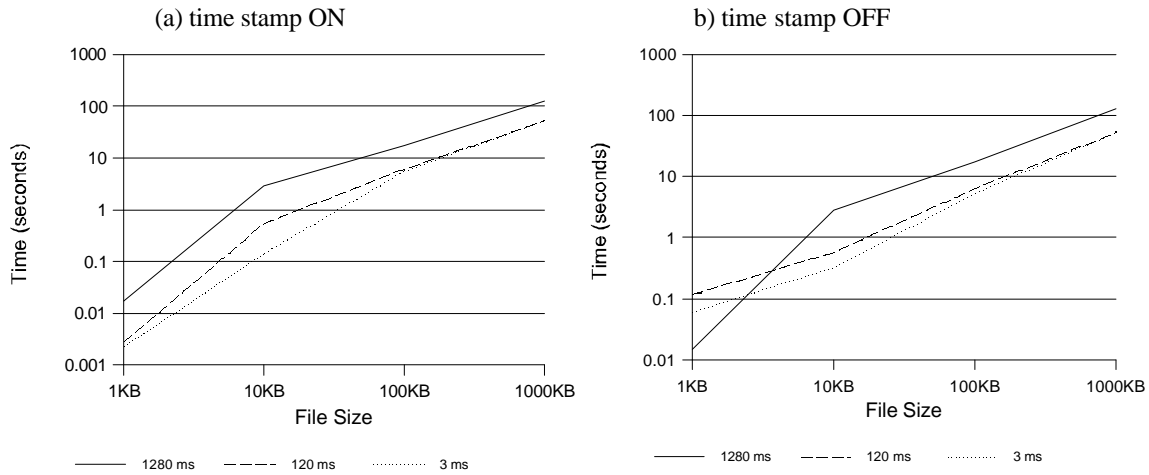


Figure 9 – Effect of time stamps on throughput for the f_p protocol using the VJ congestion control as a function of link delay.

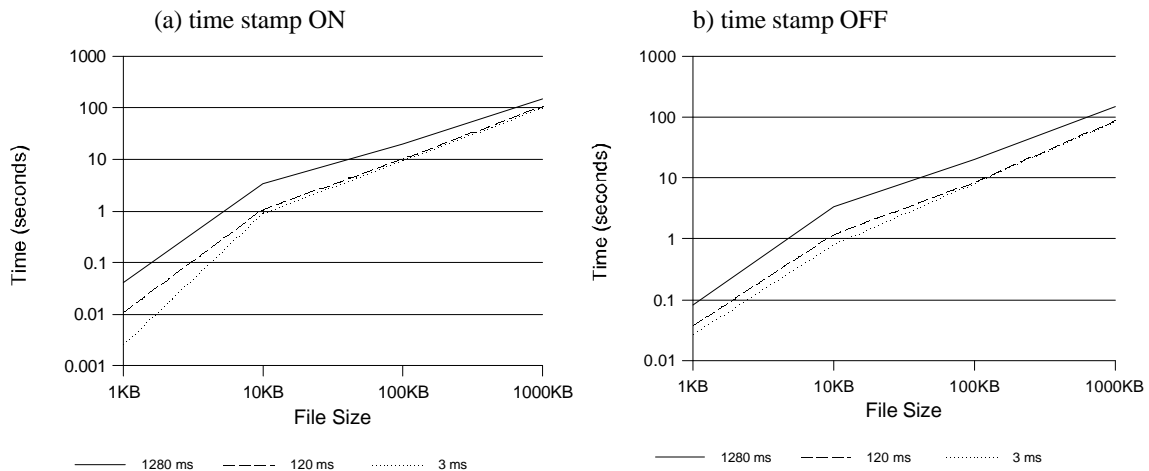


Figure 10 – Effect of time stamps on throughput for the f_p protocol using the VJ congestion control as a function of link delay for unsymmetric links.

smaller than 10 Kbytes as can be seen by the change in slope of the curves at this point. Again, we interpret this as an effect of the transmission window. Interestingly, the symmetric data links run faster with time stamps enabled than with time stamps disabled in these series of experiments for the Vegas control algorithm. As in the Van

Jacobson control algorithm there is not a major link penalty for having link delays out to GEO orbits for the large data files.

Generally, the Vegas congestion control algorithm throughput times in Figures 11 and 12 are higher than the Van Jacobson congestion control algorithm times in Figures

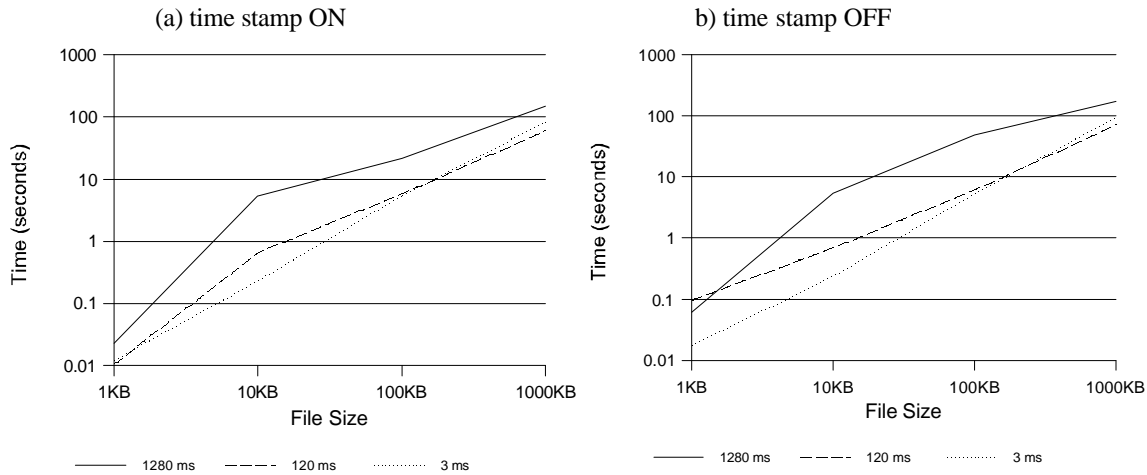


Figure 11 – Effect of time stamps on throughput for the \mathcal{F}_P protocol using the Vegas congestion control as a function of link delay for symmetric links.

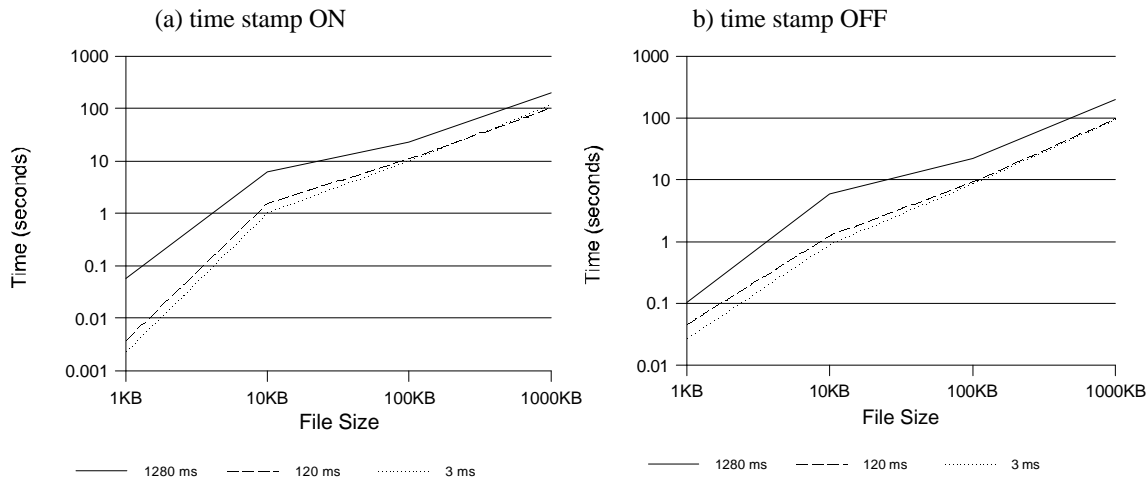


Figure 12 – Effect of time stamps on throughput for the \mathcal{F}_P protocol using the Vegas congestion control as a function of link delay for unsymmetric links.

9 and 10 for the larger file transfer experiments. This is especially true for the 3-millisecond delay times.

Effects of Header Compression

As explained in the Introduction, the TCP/IP ftp protocol can have TCP VJ header

compression enabled or disabled as a protocol option. In this set of experiments, we look at the effects of the header compression without delay in the link and with delay in the link. Because the SCPS \mathcal{F}_P protocol cannot utilize VJ header compression due to the limitations of the Linux PPP driver, SCPS will not be examined in this set of experiments.

Effect of Header Compression Without Link Delay

The first case we examine is the case where there is no link delay. Here we examine the ftp throughput as a function of channel BER and file size. Figure 13 shows the results for symmetric links with header compression enabled and disabled while Figure 14 shows similar results for the unsymmetric links. TCP VJ header compression was especially designed to improve TCP/IP performance over low-speed serial links, that is, links ranging from 300 bps to 19200 bps. Both the symmetric and unsymmetric links in our testbed should be considered to be similar low-speed serial links. Therefore, we would expect that the use of VJ header compression would improve the data throughput. For the 100-Kbyte and 1-Mbyte files transmitted over symmetric links, the improvement in throughput averages to 10% across all BER cases. For the unsymmetric links, the results are mixed and the average improvement is only 1%. The slower ACK response is interpreted as dominating over the improved header efficiency in the unsymmetric cases.

Effect of Header Compression With Link Delay

The second case we examine is the case where there is link delay. Here we examine the ftp throughput as a function of file size delay values corresponding to LEO, GEO, and lunar distances. Figure 15 shows the results for symmetric links with VJ header compression enabled and disabled while Figure 16 shows similar results for the unsymmetric links. The major effect to note is that for the large data file experiments, the link delay out to GEO orbits, does not adversely affect the data throughput over minimal delays. We attribute

this to the interactions of the operating system and the ACK process being more important for these transfers than a sun-second link delay. Generally, the use of header compression did not give a significant advantage in these experiments.

Conclusions

From this series of experiments using the simulated space channel in a point-to-point mode over a PPP data link layer with a maximum channel BER of 10^{-5} , a maximum link speed of 115200 bps, and file transfers up to 1 Mbytes in length, we make the following conclusions with respect to the TCP/IP ftp and SCPS fp protocol options:

1. Both the ftp and fp protocols perform similarly in this environment when using the Van Jacobson congestion control algorithm on both protocols. This is the standard congestion control algorithm for ftp and it is the non-default option for the SCPS-TP. Since the ftp protocol can exploit the header compression options in the PPP link under Linux while fp cannot due to software limitations in Linux/PPP, there is a slight advantage to ftp in this environment.
2. From the performance of the fp protocol in these experiments, we conclude that the Van Jacobson congestion control algorithm performs better than the Vegas congestion control algorithm under these conditions. This result is consistent with results reported in Hengartner [9]. This is the non-default option in the reference implementation and users should seriously consider this option when configuring SCPS. We attribute

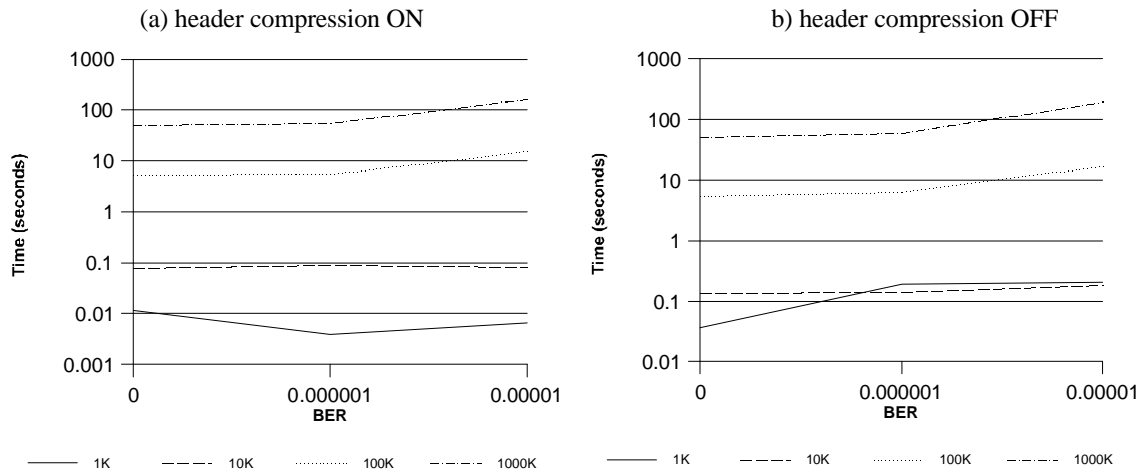


Figure 13 – Header compression effect on throughput for the ftp protocol on symmetric links as a function of file size and link BER.

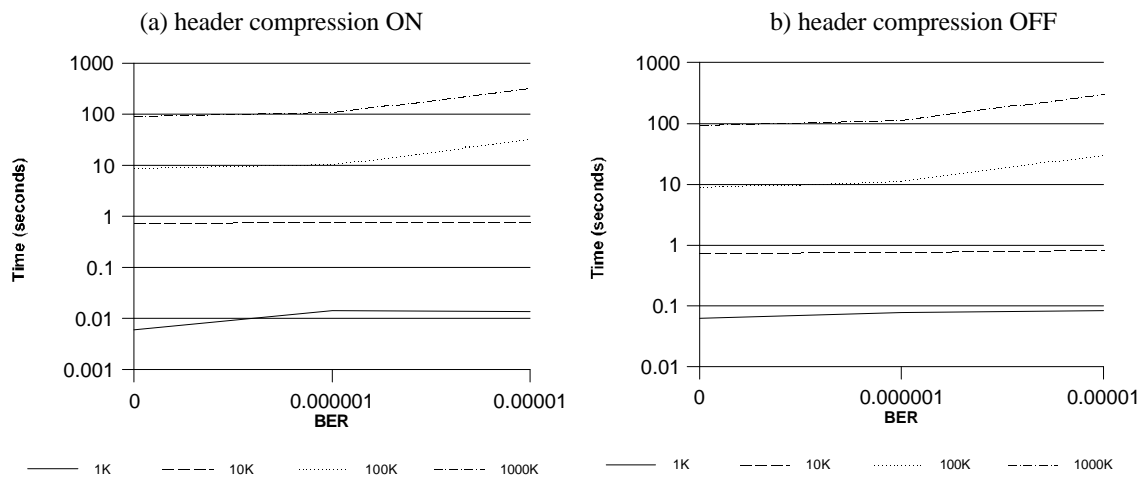


Figure 14 – Header compression effect on throughput for the ftp protocol on unsymmetric links as a function of file size and link BER.

this to the fact that the link is basically a point-to-point link with really no congestion. When packets are lost, they happen at statistically random times and not due to a network problem. Therefore, there is nothing for the Vegas algorithm to really predict. The VJ algorithm does a better job of reacting to purely

statistical occurrences not caused by networking problems.

3. For large file transfers, > 100 Kbytes, there is no major link penalty for data links out to GEO orbits over the “no delay” case. The file transfer times are nearly the same. This is attributed to the link delay being under one second in duration and the link speed,

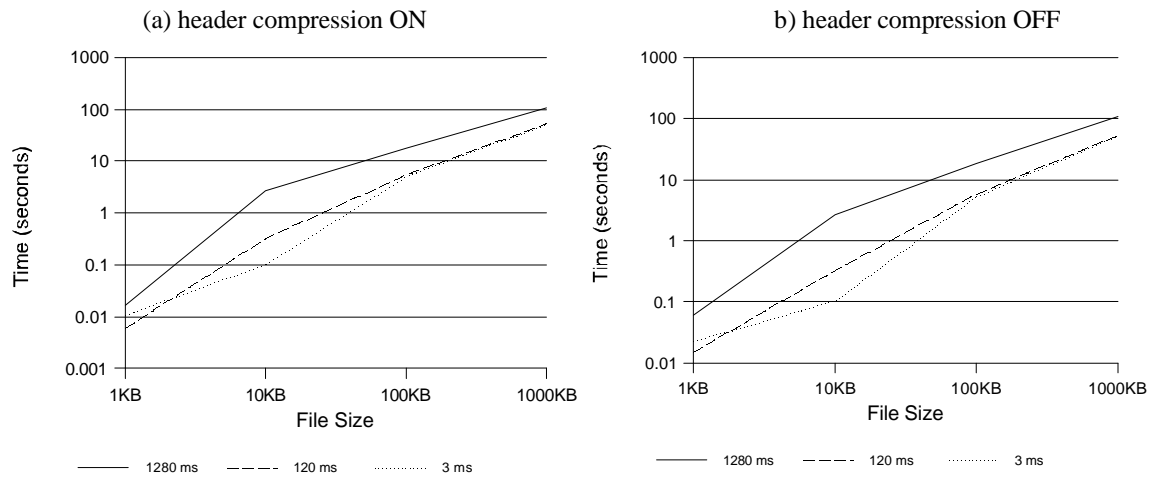


Figure 15 – Effect of header compression on throughput for the f_{tp} protocol as a function of link delay for symmetric links.

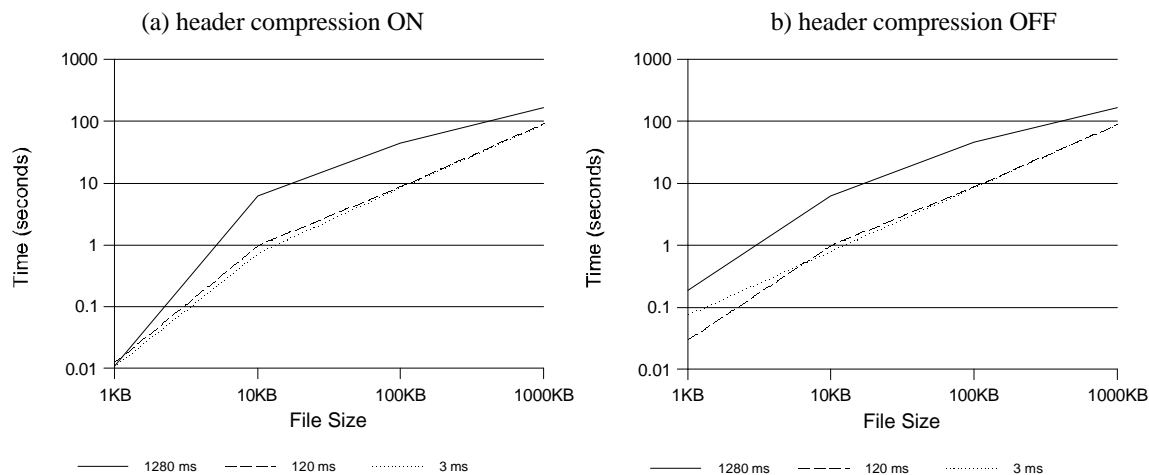


Figure 16 – Effect of header compression on throughput for the f_{tp} protocol as a function of link delay for unsymmetric links.

4. Disabling the time stamps and enabling the header compression options do provide some small transfer advantages when operating the links.

Naturally, for higher link speeds, different data link layer protocols, and different types of transactions, these results may not be valid. The intent for these experiments was to emulate the file transfer for a small satellite system over a short-duration pass to a ground station with a return link operating under 1 Mbps and a forward link operating around 2

kbps. Other conditions may dictate other protocol choices.

Acknowledgments

This research was conducted under NASA Goddard Space Flight Center research grants NAG5-7520 and NAG5-9323 to New Mexico State University. The assistance of Keith Scott and Patrick Feighery of MITRE to configure and understand the SCPS protocol is gratefully acknowledged.

References

- [1] Consultative Committee for Space Data Systems, "Space Communications Protocol Specification (SCPS) - File Protocol (SCPS-FP)," CCSDS 717.0-R-3, September 1997.
- [2] Horan, S. and Wang, R., "Design of a Channel Error Simulator Using Virtual Instrument Techniques for the Initial Testing of TCP/IP and SCPS Protocols," NMSU-ECE-99-002, 1 April 1999.
- [3] Horan, S. and Wang, R., "Enhancement of The NMSU Channel Error Simulator to Provide Unbalanced Forward And Return Transmission Rates," NMSU-ECE-99-003, April 1999.
- [4] Horan, S. and Wang, R., "Enhancement of the NMSU Channel Error Simulator to Provide User-Selectable Link Delays," NMSU-ECE-00-001, May 2000.
- [5] Stevens, W., *TCP/IP Illustrated Vol.1*, Chapter 20, Addison-Wesley, Reading, MA, 1994., p. 289.
- [6] Feighery, P., private communication, 1999.
- [7] MITRE, SCPS software release 1.1.34, 1999.
- [8] Horan, S. and Wang, R., "Effects of Protocol Options on Data Throughput Over a Simulated Space Channel," NMSU-ECE-00-007, July 2000.
- [9] Hengartner, U., Bolliger, J. and Gross Th. "TCP Vegas Revisited", INFOCOM'00.

Author Biographies

Stephen Horan received an A.B. degree in Physics from Franklin and Marshall College, an M.S. degree in astronomy in 1979, the M.S.E.E degree in 1981, and the Ph.D. degree in electrical engineering in 1984 all from New Mexico State University. From 1984 through 1986, he was a software engineer and systems engineer with Space Communications Company at the NASA White Sands Ground Terminal where he was involved with the software maintenance and specification for satellite command and telemetry systems and operator interfaces. In 1986 he joined the faculty at New Mexico State University where he is presently a Professor and holder of the Frank Carden Chair of the Telemetry and Telecommunications Program in the Klipsch School of Electrical and Computer Engineering. His research and teaching interests are in space communications and telemetry systems. Dr. Horan is the author of *Introduction to PCM Telemetry Systems* published by CRC Press. He is also a Senior Member of AIAA, a Senior Member of the IEEE, and a member of Eta Kappa Nu.

Ruhai Wang received his BS with high honors in China in 1991 and MS in Telecommunications with honors from Roosevelt University in 1997. He is currently working on his Ph.D. in Electrical

Engineering at New Mexico State University.
His research interest is on Data Networks.