SPACE COMMUNICATIONS PROTOCOL STANDARDS (SCPS)
BENT-PIPE
EXPERIMENT REPORT

SCPS D71.51-Y-1

Prepared by: _____________________________ Date:___________
Darrell E. Ernst
SCPS Flight Test Engineer

Prepared by: _____________________________ Date:___________
Robert C. Durst
SCPS Flight Test Engineer

Submitted by: _____________________________ Date:___________
STANLEY S. STEVENS
Colonel, USAF
Director for Logistics
HQ NORAD/USSPACECOM

ABSTRACT

This paper documents the results of the SCPS bent-pipe experiment. This experiment tested the SCPS Transport Protocol (SCPS-TP) over a satellite link with one-way delays on the order of 250 milliseconds and error rates from 0 to $>10^{-5}$. SCPS-TP performed well in these conditions, maintaining between 82% and 97% of maximum throughput (depending on packet size) at bit-error rates of up to 10^{-5}. We compared the performance of SCPS-TP to that of Transmission Control Protocol (TCP) using a similar configuration in the laboratory. SCPS-TP performance was equivalent to that of TCP at low bit error rates, and significantly better than TCP's at bit-error rates of 10^{-7} or greater. The two main reasons that SCPS-TP performs so much better than TCP at the higher error rates are that SCPS-TP differentiates between losses due to congestion and losses due to corruption, and that SCPS-TP has a Selective Negative Acknowledgment capability to expedite recovery from errors.

EXECUTIVE SUMMARY

PURPOSE OF THIS DOCUMENT

This report documents the findings of the SCPS Bent-Pipe test, an experiment using a DOD satellite communications package, designated M-22, to evaluate a set of communications protocols being developed by the joint NASA/DOD Space Communications Protocol Standards (SCPS) project.

BACKGROUND

In the fall of 1992, NASA and the DOD jointly established a technical team (the SCPS Technical Working Group, or "SCPS-TWG") to explore possibilities for developing common space data communications standards. By the end of 1993 the team concluded that wide segments of the U.S. civil and military space communities have common needs for protocols to support in-flight monitoring and control of civil and military spacecraft.

The program of work to develop these protocols includes specification, simulation, implementation, and testing. The SCPS Bent-Pipe Experiment is the third in a series of tests of the protocols. The first test used simulations to evaluate the protocol functional design. The second test used a laboratory test bed to evaluate the performance and functioning of prototype software that implements the protocols. The third test, the SCPS Bent-Pipe Experiment, used the M-22 system to connect two SCPS end systems together in order to evaluate the performance of one of the protocols over a real space link. Subsequent tests will host the prototype software on different spacecraft in various configurations to evaluate performance and functionality in the intended implementation environment.

After analyzing the protocol functions and other factors, the SCPS-TWG decided that the best use of satellite resources would be to focus the bent-pipe test on the transport layer. The SCPS Transport Protocol (SCPS-TP) is a derivative of Transmission Control Protocol (TCP) with modifications and enhancements intended to improve its performance in a satellite communications environment.

The SCPS Bent-Pipe Experiment was conducted by United States Space Command (USSPACECOM) Directorate of Logistics, USSPACECOM/J4 and MITRE personnel under the direction of, and with support from, the Air Force Materiel Command (AFMC) Space & Missile Systems Center Test and Evaluation office (SMC/TEO), and their contractor personnel.

M-22 TEST OBJECTIVES

The bent-pipe experiment put forth three objectives:

1. To measure the operation of SCPS-TP in a real satellite communications environment.

2. To characterize the performance of SCPS-TP.

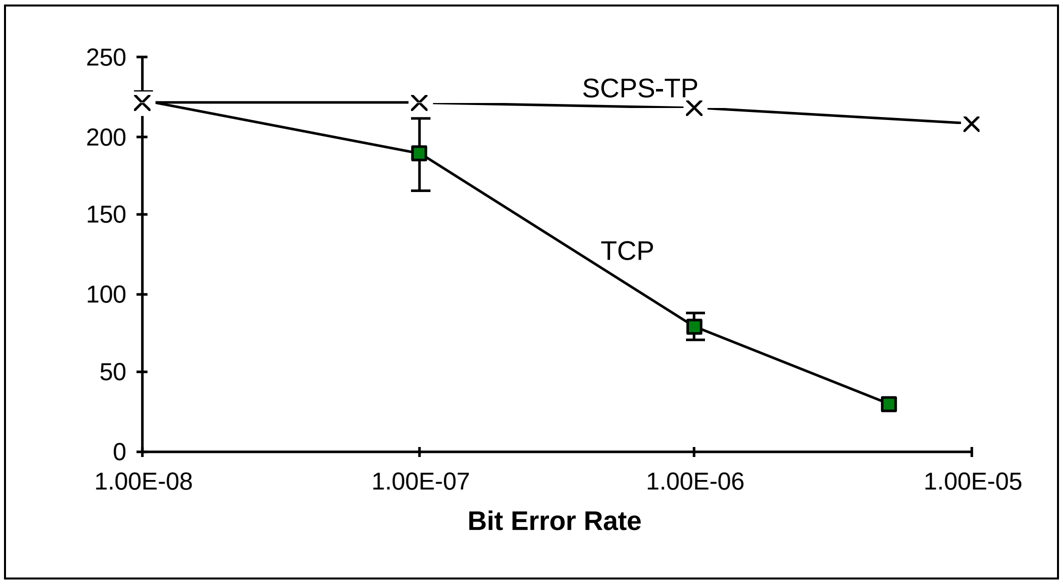
3. To compare the performance of SCPS-TP to the Transmission Control Protocol (TCP), the standard Internet reliable transport protocol, in a satellite communications environment.

Objectives 1 and 2 were met. Objective 3 was only partially met, since the tests in support of objective 3 could only be conducted in the laboratory environment, due to restrictions of some of the interface equipment used for the tests.

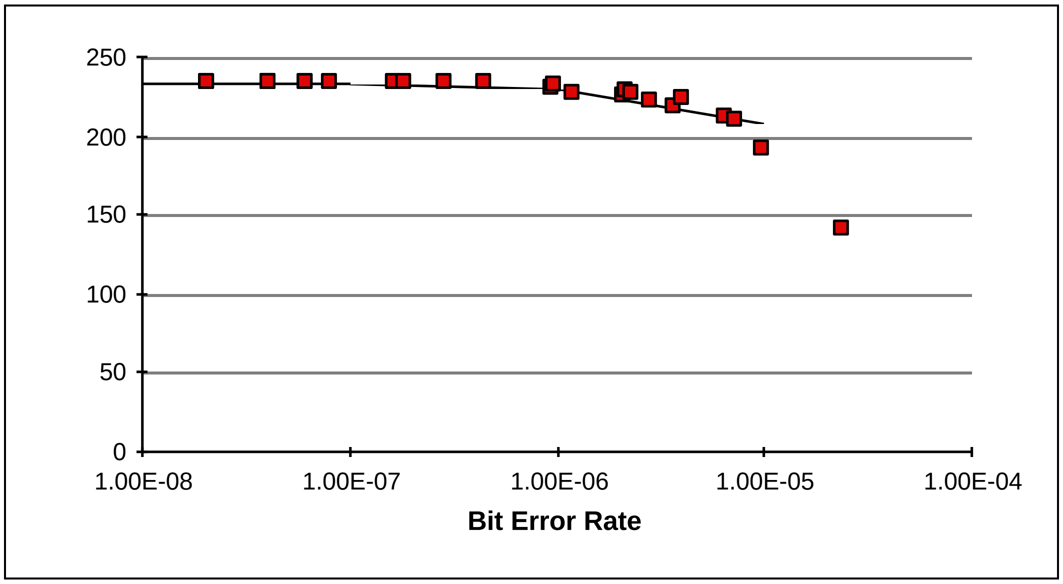
SUMMARY OF RESULTS

The experiment was conducted in two phases: we tested the protocols in the field, and then we repeated the tests in the laboratory. As a result of equipment limitations, we were unable to directly compare the performance of the SCPS-TP with TCP over the satellite link. Rather, we configured a laboratory experiment and conducted the tests there. For TCP comparison testing, we collected throughput data. For the SCPS-TP characterization data, we collected the following measurements: user data throughput, data channel utilization, volume of acknowledgment traffic generated, and bit efficiency.

The following graph compares the throughput of SCPS-TP to that of TCP for a range of bit error rates. This data was collected in the laboratory environment, with the user data per packet being 512 bytes for TCP and 500 bytes for SCPS-TP. The graph shows 90% confidence intervals around each data point. (The confidence intervals for SCPS-TP are closely-spaced, and indistinguishable from the markers for each primary data point.)



The next graph compares the SCPS-TP laboratory data to the bent-pipe experiment data. This graph shows throughput performance for 1000-byte packets. The solid line shows the laboratory data, the individual data points show the M-22 experiment data. (The laboratory data is the mean resulting from averaging five independent tests at each of the following bit-error rates: 10^{-8}, 10^{-7}, 10^{-6}, and 10^{-5}.) It should be noted that not all of the M-22 experiment data corresponds so well to the laboratory data. During the course of the experiment, we identified a number of implementation problems that affected the performance of the protocols. We have corrected most of these, but some are still under investigation. The ability to find and fix these problems was a major benefit of the bent-pipe experiment.



CONCLUSIONS

The bent-pipe experiment provided a beneficial “first step” in SCPS protocol testing. The test provided valuable insight into the performance of SCPS-TP in a real satellite communication environment. In addition, it provided the SCPS developers with an introduction to the type of ground station equipment that may be encountered during operational use of the SCPS protocols.

We are able to draw the following conclusions from this activity:

1. SCPS-TP performed equivalently to TCP in the laboratory simulation of the M-22 environment in the absence of bit errors.

2. With error rates of 10^{-7} or greater, SCPS-TP showed significantly better throughput performance than TCP (<200 Kbps vs. 30 Kbps at $5x10^{-6}$). Limitations of the particular TCP implementation we used prevented testing in the laboratory with a broader range of packet sizes. Limitations of the test hardware in Sunnyvale prevented any testing of TCP over the actual satellite link.

3. SCPS-TP performs well over a broad range of error conditions.

a) SCPS-TP throughput at a bit-error rate of 10^{-5} was measured at between 82% and 97% of the throughput in the lowest error cases. The variation was a result of different packet sizes.

b) The data channel utilization of SCPS-TP was high when the rate control parameters were properly set. In the laboratory tests, data channel utilization varied between 89% and 96%, while in the bent-pipe test it varied between 70% and 96%. In analyzing the channel utilization information, we identified problems with the rate control strategy. Some of these have been repaired, while others remain under investigation.

c) The amount of acknowledgment traffic generated was low compared to the amount of traffic the acknowledgment channel could accommodate. We improved the acknowledgment strategy to further reduce the amount of acknowledgment traffic generated; the laboratory data reflects those improvements.

d) The bit-efficiency of SCPS-TP over IP for 1000-byte and 500-byte packets met the bit efficiency goal of 10% or less protocol overhead at 0 BER, which was established in the requirements phase of the activity. Further, this goal was met for 1000-byte and 500-byte packets with error rates up to approximately $5x10^{-6}$.

4. The testing improved the quality of the SCPS-TP implementation. We made improvements to the acknowledgment strategy and to the rate control implementation as a direct result of the bent-pipe tests. We have identified areas to concentrate upon to ease deployment and field configuration of the protocol.

RECOMMENDATIONS

We offer the following recommendations as a result of the bent-pipe tests.

Broad recommendations:

1. Continue the program of SCPS testing.

   Some of the important capabilities within SCPS-TP have not yet been tested outside the laboratory - these include the Best Effort Transport Service, the User Datagram Protocol, and link outage handling. SCPS-TP should be tested in a mixed-loss environment in which losses may be due to congestion, corruption, or link outage.

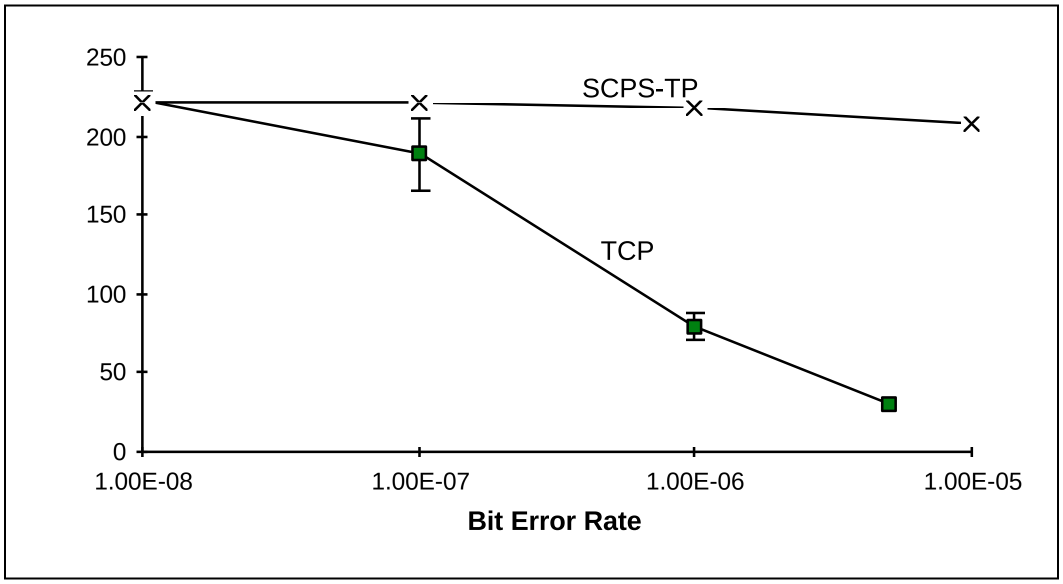
At the time we conducted these tests, we did not have the protocols fully instrumented. While we were able to collect data about the overall characteristics of each protocol test, such as throughput and amount of data transferred, we were not able to gather detailed, instantaneous data. Subsequent protocol tests should be augmented with the ability to gather information about the instantaneous dynamics of protocol operation.

2. Expand the scope of the testing to more faithfully represent operational communication requirements. The bent-pipe experiment verified the performance of SCPS-TP operating in full reliability mode, in a two-system configuration, with no other traffic present. As testing proceeds, we should attempt to identify a limited set of operational scenarios that represent more realistic uses of the SCPS protocols. Once these scenarios have been identified, we should define and execute a program of experimentation to test the SCPS protocols with these scenarios using as realistic a system configuration as possible.

Specific recommendations:

1. We recommend that further comparative testing be conducted with an implementation of TCP that permits specification of the maximum segment size. We also recommend that a test environment be identified in which TCP can be compared directly to SCPS-TP over actual satellite communication conditions and that comparative testing be conducted.

2. We do not currently have instrumentation and analysis tools to allow us to examine the instantaneous data rate on the acknowledgment channel. This is an important capability which we should develop, in order to ensure proper performance over links with restricted acknowledgment channels, and we recommend that it be developed.


250
200
150
100
50
0
SCPS-TP
TCP
1.00E-08
1.00E-07
1.00E-06
1.00E-05
Bit Error Rate

PREFACE

This report documents the findings of an experiment using a DOD satellite communications package, designated M-22, to evaluate a set of communications protocols being developed by the joint NASA/DOD Space Communications Protocol Standards (SCPS) project. The SCPS Bent-Pipe Experiment was conducted by United States Space Command (USSPACECOM) Directorate of Logistics, USSPACECOM/J4 and MITRE personnel under the direction of, and with support from, the Air Force Materiel Command (AFMC) Space & Missile Systems Center Test and Evaluation office (SMC/TEO), and their contractor personnel.

The SCPS project is developing a set of standard space data communications protocols in order to increase the interoperability and reduce the cost of space systems, with the goal of improving the delivery of space services to the end user. The users of the SCPS protocols may be working in either a civilian or military environment. The SCPS Bent-Pipe Experiment is the third in a series of tests of the protocols. The first test used simulations to evaluate the protocol functional design. The second test used a laboratory test bed to evaluate the performance and functioning of prototype software that implements the protocols. The third test, the SCPS Bent-Pipe Experiment, used the M-22 system to connect two SCPS end systems together in order to evaluate the performance of one of the protocols over a real space link. Subsequent tests will host the prototype software on different spacecraft in various configurations to evaluate performance and functionality in the intended implementation environment.

The M-22 satellite communications package includes Space-Ground Link System (SGLS) transponders and flies on a variety of U.S. spacecraft in highly ellipitcal orbits. This geometry provided an ideal means of exercising the SCPS transport protocol over the full range of the space environment for which the DOD intends to use it.

Refer to the Bibliography and List of References for references to relevant documentation.

ACKNOWLEDGMENTS

The authors would like to thank the Commander, Detachment 2, SMC, and Loral Space & Range Systems, Sunnyvale, CA, for kindly allowing us to incorporate their *SCPS Support Report* into this report. Their report is reproduced in its entirety, without modification, as an appendix. It has been included because it contains technical information that the authors deem crucial to the understanding of the experiment, and to help users of this report to make independent judgments about the findings reported herein. The authors feel that the Detachment 2 report would be difficult to obtain by most readers if it were incorporated merely by reference.

TABLE OF CONTENTS

SECTION			PAGE
1.0	Introduction		1
2.0	Experiment Objectives		3
3.0	Experiment Design		4
4.0	Experiment Configuration		5
	4.1	Laboratory Configuration	5
		4.1.1 SCPS Test Bed Equipment Configuration	5
		4.1.2 TCP Protocol Tests	6
		4.1.3 SCPS-TP Tests	8
	4.2	Bent-Pipe Configuration	12
		4.2.1 Computer Equipment Configuration	12
		4.2.2 Bit-Error Rate Test Equipment Configuration	15
		4.2.3 Cryptographic Equipment Configuration	16
		4.2.4 Bent-Pipe Test Data Flow	16
		4.2.5 SCPS-TP Protocol Configuration	16
5.0	Experiment Methods		20
	5.1	Data Collection	20
	5.2	Error Introduction and Measurement	20
6.0	Experiment Results		21
	6.1	Laboratory Experiment	21
		6.1.1 User Data Throughput Results	21
		6.1.2 Data Channel Utilization	24
		6.1.3 Acknowledgment Traffic	26
		6.1.4 Bit Efficiency	28
	6.2	Bent-Pipe Experiment	30
		6.2.1 User Data Throughput Results	30
		6.2.2 Data Channel Utilization	37
		6.2.3 Acknowledgment Traffic Results	43
		6.2.4 Bit Efficiency Results	46
	6.3	Link Quality Analysis	50

7.0 Conclusions and Recommendations 62

TABLE OF CONTENTS

SECTION **PAGE**

Bibliography and List of References 65

Appendix A Data 67

Appendix B Equations 89

Appendix C SCPS Support Report 95

Appendix D SCPS Protocol Overview 109

Appendix E SCPS-TP Implementation Background 119

Glossary 123

Distribution List 125

LIST OF FIGURES

FIGURE		PAGE
1	Laboratory Test Configuration for TCP	7
2	Laboratory Test Configuration for SCPS-TP	9
3	SCPS Bent-Pipe Experiment Configuration	14
4	Bit Error Rate Test Setup	15
5	SCPS Bent-Pipe Experiment Data Flow	19
6	SCPS-TP and TCP Throughput versus Bit Error Rate	23
7	SCPS-TP Throughput versus Bit Error Rate for Various Packet Sizes	24
8	User Data Vesus Total Data Use of Data Channel for 1000-byte Packets	25
9	User Data Versus Total Data Use of Data Channel for 500-bype Packets	26
10	Effect of Acknowledgment Rate on Acknowledgment Data Sent	27
11	Effect of Acknowledgment Rate of User Data Throughput	28
12	Bit Efficiency of SCPS-TP Over IP	29
13	SCPS-TP Throughput Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data	30
14	SCPS-TP Throughput Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data	32
15	SCPS-TP Throughput Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data	34
16	SCPS-TP Throughput Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data	36
17	SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Date	38

LIST OF FIGURES (continued)

FIGURE **PAGE**

18 SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data 40

19 SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data 41

20 SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data 42

21 SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data 43

22 SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data 44

23 SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data 44

24 SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data 45

25 SCPS-TP Bit Efficiency Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data 47

26 SCPS-TP Bit Efficiency Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data 47

27 SCPS-TP Bit Efficiency Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data 48

28 SCPS-TP Bit Efficiency Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data 48

29 Bit Error Rate Derived from Laboratory Data 53

30 Derived versus Measured Errors, 5 December 1995 54

31 Derived versus Measured Errors, 6 December 1995 55

32 Derived versus Measured Errors, 7 December 1995 56

LIST OF FIGURES (concluded)

FIGURE		PAGE
33	Derived versus Measured Errors, 8 December 1995	57
34	Derived versus Measured Errors, 12 December 1995	58
35	Derived versus Measured Errors, 13 December 1995	59
36	Derived versus Measured Errors, 14 December 1995	60
37	Derived versus Measured Errors, 15 December 1995	61

LIST OF TABLES

TABLE		PAGE
1	SCPS-TP Configuration Information for Laboratory Experiments	11
2	SCPS-TP Configuration for Bent-Pipe Experiment	17
3	Link Measurement Summary Data	51

1.0 INTRODUCTION

In the fall of 1992, NASA and the DOD jointly established a technical team (the SCPS Technical Working Group, or “SCPS-TWG”) to explore possibilities for developing common space data communications standards, with a principal focus on the activities associated with in-flight monitoring and control of civil and military spacecraft. In practical terms, these activities involve a ground control center conducting a dialog with a remote spacecraft to transmit telecommands, to up-load and verify onboard software loads, and to confirm correct spacecraft performance via a flow of telemetry.

The team adopted a two-pronged approach in its study phase: part of the team conducted a top-down survey of representative civil and military space data communications requirements, while the remainder of the team conducted a bottom-up analysis of available standard data communications protocols. The team compared the results to see how capabilities matched requirements, and formulated recommendations for future work. In evaluating existing capabilities, first priority was given to commercially-supported “off the shelf” standards. However, recognizing unique requirements of the space mission environment (long propagation delays, noise-induced errors, and limited spacecraft data processing resources and communications capacity), other options were also considered. By the end of 1993 the team concluded that wide segments of the U.S. civil and military space communities have common needs for:

- An efficient file handling protocol, capable of supporting file transfers initiated either from ground-based systems or space-based systems
- A data transport protocol that provides the user with selectable levels of reliability, based on operational need, between computers that are communicating over a network containing one or more space data transmission paths
- Optional data protection mechanisms to assure the end-to-end security and integrity of such message exchange
- An efficient protocol to support connectionless routing of messages through networks containing space data links.

Following the study phase, the SCPS-TWG began development of four specifications, one for each of the protocols that addresses the above requirements: the SCPS File Protocol (SCPS-FP), the SCPS Transport Protocol (SCPS-TP), the SCPS Security Protocol (SCPS-SP), and the SCPS Network Protocol (SCPS-NP). At the completion of the development phase, the specifications will be submitted to the appropriate DOD authority for adoption as military standards, and to the appropriate international body for consideration of adoption as international standards. These standards may then be adopted by any military, civil, or commercial organization for use in any space system. It is the intent of NASA and DOD that commercial vendors produce the SCPS protocols as widely-distributed commercial products, thus helping to reduce the cost of space systems while increasing their interoperability.

Part of the development program includes a rigorous test program. As the protocol specifications are developed, the design engineers develop software that implements the various features to help analyze the design. Eventually this software represents a full implementation of the protocol. This software is run in a simulation of a communications environment to help fine-tune the protocol design. When the designer feels the design is sufficiently stable, the software is installed on a communications test bed that includes a large number of physically separate communications nodes and a number of space environment simulators. This test bed may be configured into a large number of network topologies for the purpose of testing the various capabilities of the SCPS protocols.

The bent-pipe test program provided for a test of the communication-path-sensitive protocols using an actual satellite link after testing in the test bed. This test required two workstations running the protocols to be connected together via a “bent-pipe” satellite link; i.e., the satellite would provide a simple repeater at the physical layer between the workstations without any processing aboard the spacecraft. After analyzing the protocol functions and other factors, the SCPS-TWG decided that the best use of satellite resources would be to focus the bent-pipe test on the transport layer.

Initial testing of the SCPS-TP in the test bed was completed in November 1995. The prototype software was prepared for the bent-pipe experiment and installed on the experiment workstations in mid-November. After a period of checkout, the experiment formally started on 4 December 1995. Although attempts were made to conduct test runs until 22 December, the last set of acceptable test data was recorded on 15 December 1995. For reasons discussed below, the experiment followed a modified version of the experiment design described in [1].

The bent-pipe experiment was conducted using an US Air Force satellite special mission package that flies on classified US satellites. The mission package is referred to by various names, including M-22, an older designation, and M2P1, the designation for the mission packages used on this experiment. This report will use the M-22 designation. The use of the packages by non-prime mission users is overseen by the M-22 User’s Group who review the experiment applications and make recommendations for approval or disapproval to the M-22 Steering Group.

The two satellites used for this experiment are in highly elliptical orbits that provide very long view times over specific parts of the earth. The specific orbital parameters were not germaine to the experiment and are not included in the report. The tracking information necessary to support data analysis was recorded and is included in the report. The combination of the satellite orbits and the M-22 communications package provided an almost ideal environment for the protocol test: the delays, long view times, transponder data rate capacity, presence of signal obscurations, and the ability to artificially degrade the link were all desirable elements of the experiment.

This document is organized into eight sections and five appendixes. Section 1 is this Introduction. Section 2 puts forth the experiment objectives. Section 3 describes the experiment plan and Section 4 describes the experiment configuration. Section 5 documents the experiment methods that were used to conduct the experiments and gather the data. Section 6 presents the experiment results, and Section 7 presents conclusions and recommendations. Appendix A contains the data collected during the experiment. Appendix B documents the equations used in the data analysis. Appendix C reproduces in its entirety the "SCPS Support Report" produced by Loral Space and Range Systems. Appendix D presents an overview of the SCPS-TP protocol, and Appendix E describes the SCPS-TP implementation.

2.0 EXPERIMENT OBJECTIVES

The objectives of the SCPS bent-pipe experiment were as follows:

1. To measure the operation of SCPS-TP in a real satellite communications environment.

2. To characterize the performance of SCPS-TP.

3. To compare the performance of SCPS-TP to the standard Internet protocol suite, TCP.

The first objective is concerned with the basic operation of the protocol in a high-delay environment, compared to terrestrial communication, with varying bit error rate conditions.

The second objective is concerned with the overall performance of the protocol with a number of different packet sizes, with varying error conditions. The basic measure of performance is quantity of data delivered to the user per unit time, but other performance factors, such as bit efficiency, are needed for supporting the analysis. These are discussed below.

The third objective, although not explicitly described in [1], is intended to provide the information necessary to address legitimate concerns raised by members of the interested community, to wit; will TCP provide the required capabilities and thus obviate the need for a new protocol.

In the original experiment design, two experiments addressed these three objectives: a confirmation experiment and a characterization experiment. These experiments and a third experiment, the verification experiment, are described in Section 3. Circumstances did not permit the execution of these experiments as originally designed, and the modifications to the original experiment design are also discussed in Section 3.

3.0 EXPERIMENT DESIGN

The actual bent-pipe experiment deviated from the planned experiment that is documented in reference [1]. The significant deviations are described below.

The original plan called for a comparison of TCP and SCPS-TP over the M-22 satellite link. For a reason yet to be determined, the TCP implementation would not operate through the high-speed serial interface cards used on the workstations to provide connectivity to the satellite ground system. Numerous attempts to obtain technical support from the manufacturer went unanswered. Since this directly related to objective 3 of the experiment, we tested TCP and SCPS-TP in the laboratory, with the laboratory environment configured to simulate the delays and errors of the M-22 bent-pipe test.

One of the experiment design factors was link outage. It turned out that the particular satellite ground station equipment used could not provide the necessary signal to allow the protocol to know when the link was out. Further, the workstation interface cards had no way to appropriately consume that signal and pass it to SCPS-TP if it were provided. As a consequence, this factor was eliminated from the design.

The precise control of link bit error rate (BER) could not be achieved. The use of attenuators was not considered feasible because of the undesirability of attenuating both the signal and the noise. The method chosen for BER control was to reduce the uplink modulation index (see Appendix C). As it turned out, there was no precise way to calibrate the modulation index control to be able to consistently set a predetermined BER for a given base (unmodified) BER. Consequently, we used the following approach to approximate the desired bit error rate:

1) measure the unmodified BER,
2) adjust the modulation index to “low,” “moderate,” or “high” settings as appropriate,
3) verify the setting with a BER test (BERT) to determine the error count, and finally,
4) execute the protocol tests.

The original experiment design contained two settings for the packet size factor. Subsequent to the issuance of the experiment plan there was some supposition that the protocol response would not vary linearly as the packet size changed. Since the number of data runs had been reduced for the reasons described above, it was decided to increase the number of packet size settings from two to four.

As a result of the modifications to the experiment design, we conducted two sets of tests: the M-22 bent-pipe tests, conducted using the M-22 satellites; and laboratory tests, in which we simulated the delays, errors, and data rates of the actual M-22 test environment. We compared the performance of SCPS-TP to that of TCP in the laboratory tests, and we characterized the performance of SCPS-TP in the M-22 bent-pipe tests.

4.0 EXPERIMENT CONFIGURATION

We performed the M-22 experiments in the laboratory, using the SCPS test bed, and in the field, using equipment provided by AFMC at Loral's facility in Sunnyvale, CA. This section describes the experiments conducted in each of those two locations.

4.1 LABORATORY CONFIGURATION

We tested both TCP and SCPS-TP in the laboratory, using a test configuration that simulated the delays, data rates, and error rates of the field environment. The laboratory equipment was generally the same for the TCP and SCPS-TP tests, but the test configurations of the supporting equipment differed. (The reason for the differences stems from the fact that the TCP implementation resides within the operating system kernel, while the SCPS-TP implementation currently operates above the operating system kernel.)

This section describes the SCPS test bed equipment that was used in the tests, then describes each test separately. For each test, we describe the test bed configuration and data flow, the protocol configuration, and the test driver programs.

4.1.1 SCPS Test Bed Equipment Configuration

The laboratory testing required three workstations: a data source (the initiator), a data sink (the responder), and a system to emulate the bent-pipe test environment's delays, errors, and data rates. All systems involved in the test bed experiments were interconnected by a single Ethernet LAN segment.

4.1.1.1 Initiator and responder systems

We used Sun Sparc workstations for the initiator and responder systems. We used a combination of Sparc ELC and Sparc 20 workstations, all running SunOS 4.1.3. The data rates in the bent-pipe test were sufficiently low enough that all of the Sparc workstations in the test bed were much "faster" than the network. That is, the processor power of the workstations did not constitute a bottleneck resource in either the TCP tests or the SCPS-TP tests.

4.1.1.2 Delay and error emulation - the Spanner program

A third Sun Sparc workstation hosted a program, called Spanner, that was used to emulate the delays and errors of space communication. The system hosting Spanner was connected to the initiator and responder systems via a (single) Ethernet local area network (LAN). The following paragraphs briefly describe the operation of the Spanner program:

The Spanner program emulates the errors encountered in spacecraft communication. We use a Bernoulli sequence to determine the probability that a packet will be corrupted. The length of the packet and the user-supplied bit error rate determine the probability that a packet will be corrupted in transmission. Packets that are “corrupted” are simply not forwarded to the destination.

The Spanner program uses two components to emulate space communication delays. The first component is a fixed value that represents the propagation delay due to the distance travelled at the speed of light. For the bent-pipe test, the one-way fixed delay was estimated to be 250 milliseconds from initiator to responder and 250 milliseconds from responder to initiator. The second component is variable, emulating the delays imposed by the data rates experienced from initiator to responder or from responder to initiator. The Spanner program calculates the queuing delay experienced by the packet based on the data rate and the amount of data queued for transmission. When a packet is received by the Spanner program, these two components of delay are computed for the packet, and its departure time is scheduled. (Note that packets that are affected by bit errors are still factored into the delay calculation, since they consume queuing resources before they experience the errors.)

The Spanner program represents a system with essentially infinite queuing at the entry to the space segment. This differs from the way an actual space-link interface would operate, in that an actual system interface would have smaller input buffers and would (probably) discard incoming packets if its input buffers were full. The effect of this difference is that Spanner can build a larger input queue, and packets can incur significant amounts of queueing delay if the instantaneous arrival rate to the Spanner program exceeds the capacity of the space link.

4.1.2 TCP Protocol Tests

The TCP tests used the version of TCP shipped as part of the SunOS 4.1.3 kernel. In order to route the data through the Spanner program, rather than being forwarded directly to the destination, we were forced to reconfigure the initiator and responder. This section describes the system configurations, the resulting data flow, the protocol configuration, and the test drivers used.

4.1.2.1 Testbed configuration and data flow

Figure 1 illustrates the test bed configuration used for the TCP testing.

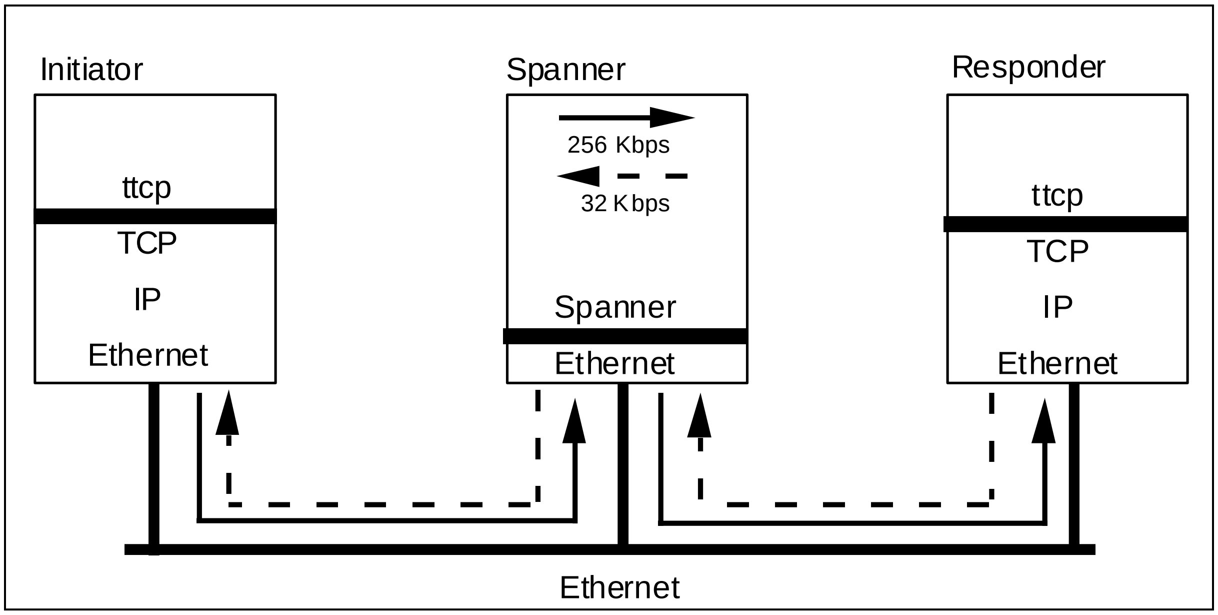


Figure 1. Laboratory Test Configuration for TCP

In order to route data from the end systems through Spanner, rather than directly to the destination, we were forced to change the IP addresses of the end systems so that they appeared to be on different (fictional) subnetworks. We assigned routing table entries in each end system that directed traffic to the other end system’s subnetwork through Spanner. The Spanner program used for the TCP tests transmits and receives data directly via the Ethernet interface. In this manner, we can use the Spanner program on in-kernel protocols without having to modify the kernel on the Spanner system (or either of the end systems).

The approach used to route data through Spanner has side-effects. The initiator and responder are configured to appear to be on different subnetworks. Most implementations of TCP, including the one in SunOS, restrict the maximum size of TCP segments to 512 bytes when the end systems are on different subnetworks. This limits the amount of user data per TCP segment to 492 bytes. Additionally, the test end systems are essentially unreachable from other workstations, since Spanner is not a general purpose router. As a result, data must be taken locally at each end system, rather than being coordinated from a single test workstation.

4.1.2.2 Protocol configuration

Our ability to configure the SunOS TCP is limited. As mentioned above, the maximum segment size is fixed at 512 bytes. The only other relevant configuration option available is the size of the socket buffer, which defines the retransmission and reassembly buffer sizes. The maximum buffer size available under SunOS 4.1.3 is 51968 bytes. Is this sufficient? At a data rate of 256000 bits per second (or 32000 bytes per second) the round-trip delay is approximately one-half second, which means that, in the absence of errors, there will be approximately 16000 bytes of data “in flight.” The maximum buffer size is somewhat over three times this amount, which is sufficient to prevent TCP from slowing down due to a lack of buffer space.

4.1.2.2 Test drivers

To generate traffic and measure the performance of Standard TCP, we used the ttcp traffic generation tool. This utility simply creates a process on each end system and transfers data over a TCP connection. It was developed at the U. S. Army's Ballistics Research Laboratory (BRL), modified by Silicon Graphics, and is widely used for conducting benchmarks. Ttcp gives the user control over some TCP parameters, such as the sender and receiver socket buffer sizes (mentioned above), which determine the TCP window size. The user may also specify the size of the writes, in bytes, that ttcp uses on the write system calls. We conducted these tests using data transfers of six million bytes. We configured ttcp to perform 6000 1000-byte writes.

4.1.3 SCPS-TP Tests

The SCPS-TP prototype has been developed as part of the SCPS standardization efforts. Unlike TCP, we elected to implement the protocol outside the Unix kernel. This simplified development and testing, but requires a slightly different laboratory configuration than that used for the TCP testing.

4.1.3.1 Testbed configuration and data flow

Figure 2 illustrates the test bed configuration for the SCPS-TP testing.

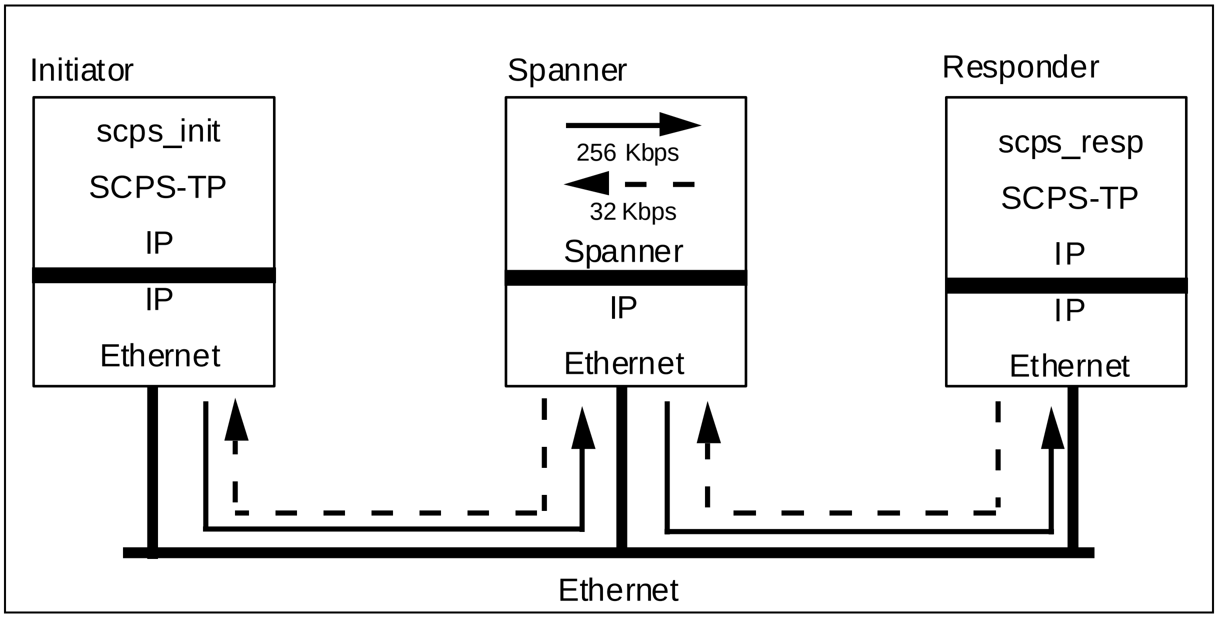


Figure 2. Laboratory Test Configuration for SCPS-TP

For the SCPS TP tests, the Spanner program used the socket interface to IP, rather than the Ethernet interface, as used for the TCP testing. The bent-pipe tests used IP as the network layer protocol. The destination address in the IP packet is that of the remote end system. In order to send these SCPS-TP/IP packets to Spanner, and then from Spanner to the remote end system, they are encapsulated in another IP packet, containing the IP address of the host on which Spanner is operating.

4.1.3.2 Protocol configuration

As a development prototype intended to be used in many different environments, the SCPS-TP protocol offers many more configuration options than the in-kernel TCP. For an overview of SCPS-TP, refer to Appendix D of this document. The configuration options that were relevant to the bent-pipe test are listed below:

- Buffer sizes
- Transmission rate control
- Optional use of congestion control

- Acknowledgment frequency control

- Use of Selective Negative Acknowledgment option
- Use of TCP Window Scaling option
- Use of TCP Timestamps option

The buffer sizes used for SCPS-TP transmit and receive buffers were 290k bytes per buffer.

We have been experimenting with transmission rate control within SCPS-TP. The routing structure contains information about the maximum data rate that can be sustained between the initiator and the responder. This information is currently statically configured, but we are investigating an adaptive mechanism based on the congestion control method defined in TCP Vegas, ref [9].

SCPS-TP congestion control was disabled for this experiment. This is appropriate when a network's bandwidth is managed to ensure that congestion is not present (typical of most current NASA and DOD missions). The effect of this configuration option also corresponds to a network in which there is explicit signalling of congestion and corruption, and losses are due to corruption.

The SCPS-TP controls the rate at which acknowledgments are sent, in order to avoid oversubscribing the capacity of the acknowledgment channel. (Many satellite links are configured with a high-rate channel in one direction and a much lower rate channel in the opposite direction. These are typically for telemetry and commands, respectively. If a satellite is reliably transmitting telemetry, the acknowledgment traffic generated by TCP can easily overrun the channel carrying the acknowledgments.) TCP requires that an acknowledgment be generated for at least every other TCP packet received. SCPS-TP removes this requirement, and generates acknowledgments much less frequently. SCPS-TP normally delays acknowledgments by a configurable amount of time. (Eventually, this delay will be tied to SCPS-TP's estimate of the round trip time, but for the purposes of experimentation, we prefer to have direct control over the parameter for the present.) In the laboratory, we examined acknowledgment rates of one acknowledgment per round trip time and two acknowledgments per round trip time.

The Selective Negative Acknowledgment (SNACK) option was defined in SCPS-TP to improve throughput performance in the presence of communication errors. Refer to Appendix D for a description of the option. We used the SNACK option in all of the laboratory tests.

The TCP Window Scaling option allows transport endpoints to have more than 64k bytes of data outstanding at one time. The retransmission and reassembly buffers, as mentioned above, were approximately 290,000 bytes. In order to inform the remote side of the receive buffer size, we used the TCP Window Scaling option in all of the laboratory tests.

The TCP Timestamps option permits accurate estimation of round trip times. The accuracy of these estimates is important when errors are present, since the round trip time determines the retransmission timer values. We used TCP Timestamps option in all of the laboratory tests.

The SCPS Header Compression option reduces the size of the TCP headers by approximately 50%. Its use reduces the size of SCPS-TP packets. This reduction is relatively insignificant for long data packets, but becomes significant for acknowledgments. When the data rate for the acknowledgment channel is highly restricted, SCPS Header Compression can permit more acknowledgments to flow over the channel. We did not use SCPS Header Compression in the laboratory experiments.

The following table summarizes the SCPS-TP configuration information used for the laboratory experiments:

Table 1. SCPS-TP Configuration Information for Laboratory Experiments

Configuration Parameter	Laboratory Setting
Buffer Size Send buffer size Receive buffer size	 297,713 bytes 289,521 bytes
User data per packet	1000 500 250 125
Rate control settings Initiator-to-Responder Responder-to-Initiator	 256 Kbps 16 Kbps
Congestion control	Off
Ack Frequency	1 ack/RTT & 2 acks/RTT
Selective Negative Acknowledgment	On
Window Scaling	On
TCP Timestamps	On
SCPS Header Compression	Off

4.1.3.3 Test drivers

We developed a simple data-transmitting client (scps_init) and a data-receiving server (scps_resp) for use in debugging, testing, and evaluating SCPS-TP. SCPS-TP has a socket-like interface, but at this time, we have not implemented an interface that is *identical* to the socket interface. Therefore, applications such as ttcp, described above, would require modification to use over SCPS-TP. It was more expeditious to develop scps_init and scps_resp, which are functionally equivalent to ttcp (the test driver used for the TCP tests).

4.2 BENT-PIPE CONFIGURATION

The laboratory configuration described above is intended to simulate the experiment configuration described in this section.

4.2.1 Computer Equipment Configuration

4.2.1.1 Initiator and responder systems

As in the laboratory configuration, we used Sun Sparc workstations for the initiator and responder systems. We used a Sun Sparc IPC workstation for the initiator, and a Sun Sparc 2 workstation for the responder. (The Sparc 2 was loaned to us for the experiment by AFMC, for which we are grateful.) These workstations are identified as SCPS Node 1 and SCPS Node 2 in Figure 3. Both workstations ran SunOS 4.1.3. Rather than using Ethernet interfaces, as in the laboratory configuration, we used RS-449 interfaces with configurable data rates to connect the workstations to the KG-94s. The workstation-resident interface cards, Sun High-Speed Interface, S-Bus (HSIS) cards, were a source of some configuration difficulties. We were unable to configure the in-kernel TCP to access these interface cards. (Telephone consultation with Sun Microsystems did not yield any solutions to this problem. Sun has a more recent product that is advertised to provide that interface, but we did not have access to it for the test.)

SCPS Node 1 and SCPS Node 2 hosted the SCPS protocol stack for the experiments. The SCPS TP protocol used IP as a network layer, and operated directly over the HSIS cards. A detailed description of the interface between the HSIS cards and the satellite ground station equipment is presented in Appendix C.

4.2.1.2 Communication interfaces and equipment

The SCPS test team was not physically resident in Sunnyvale for the execution of the tests. Rather, to minimize travel costs, we configured the test equipment in Sunnyvale, developed some simple procedures for the Loral personnel supporting the testing, then conducted the tests remotely. We used two Internet connections, one to SCPS Node 1 and one to SCPS Node 2, to communicate with the Sunnyvale equipment. When powered on, the SCPS Nodes were configured to dial the local Internet provider and connect to it. We then used the Telnet protocol over the Internet to log in to the SCPS Nodes and conduct the tests. We used voice lines to coordinate with the Loral operators to arrange bit error rate tests and other such activities.

At one point during the testing, the Internet provider experienced a severe power outage. This interrupted their service for over two days. Rather than suspending the testing, the Loral personnel helped us by executing the tests for us, taking keystoke-by-keystroke direction over the voice lines. This was a significant benefit to us.

The SCPS team conducted the tests from both Colorado Springs, CO, and from Reston, VA. Since the SCPS protocols were hosted in the SCPS Nodes, the difference in Internet connectivity made no difference.

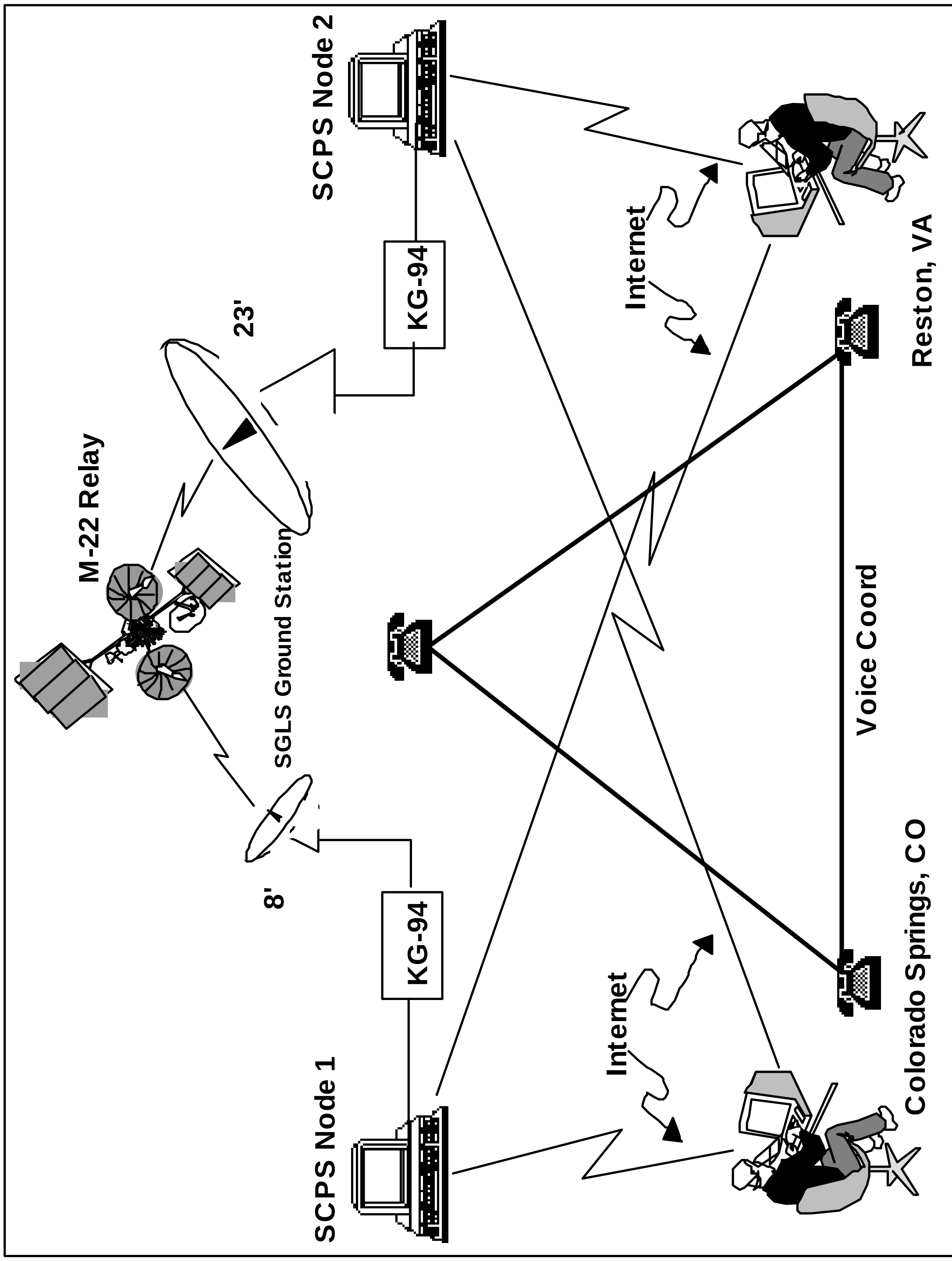


Figure 3. SCPS Bent-Pipe Experiment Configuration

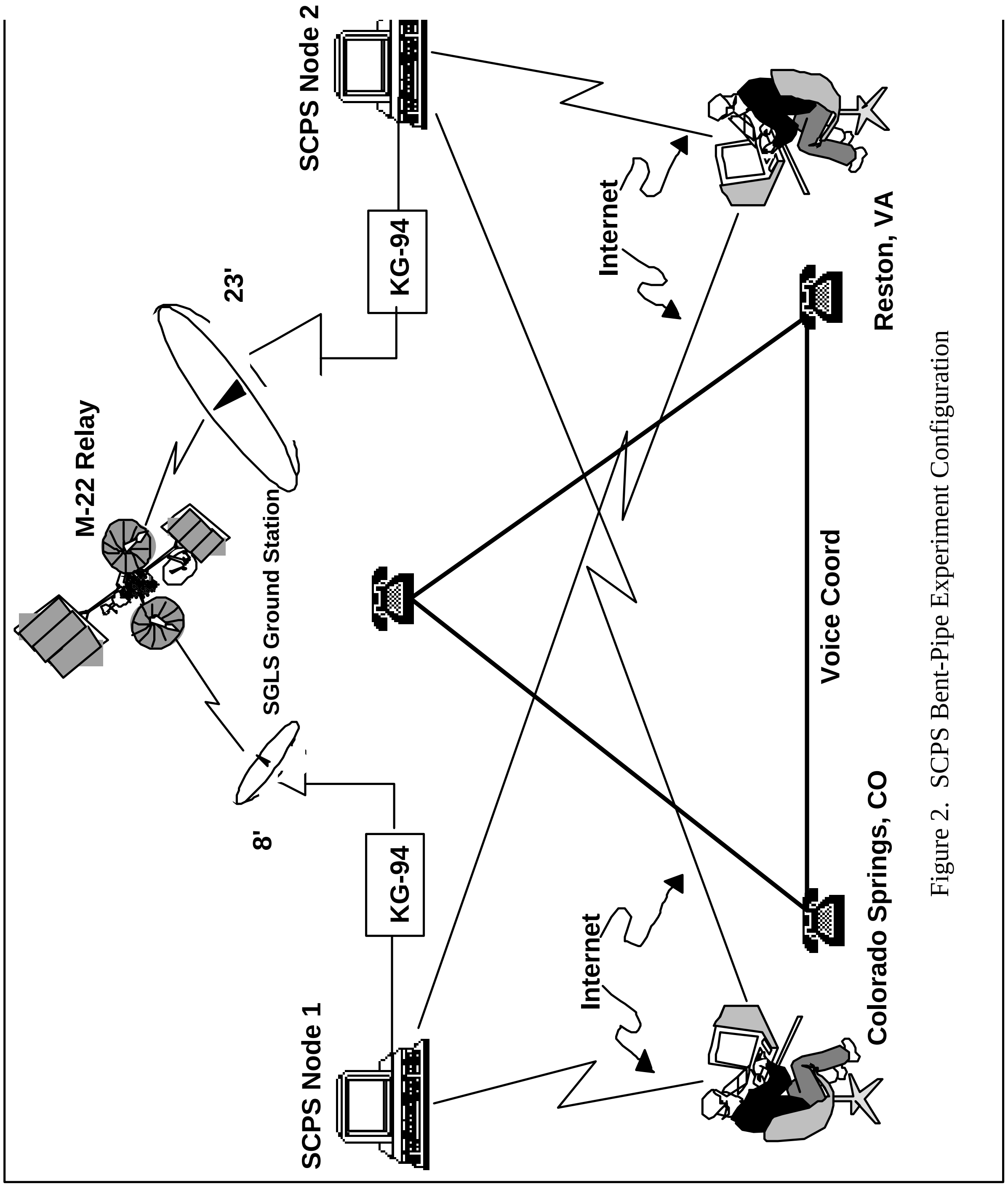


Figure 2. SCPS Bent-Pipe Experiment Configuration

4.2.2 Bit-Error Rate Test Equipment Configuration

A *Fire BERT 6000* bit error rate test set was used to perform all of the bit error tests on the satellite links. The test set was used to make a test on the low-data-rate link at the beginning of contact, and then was patched into the high-data-rate link. After checkout of that link, the SCPS workstations were patched into the respective links. The test set was left configured for the high-data-rate link for the remainder of the contact to allow for rapid testing in between data runs. Figure 4 illustrates the setup.

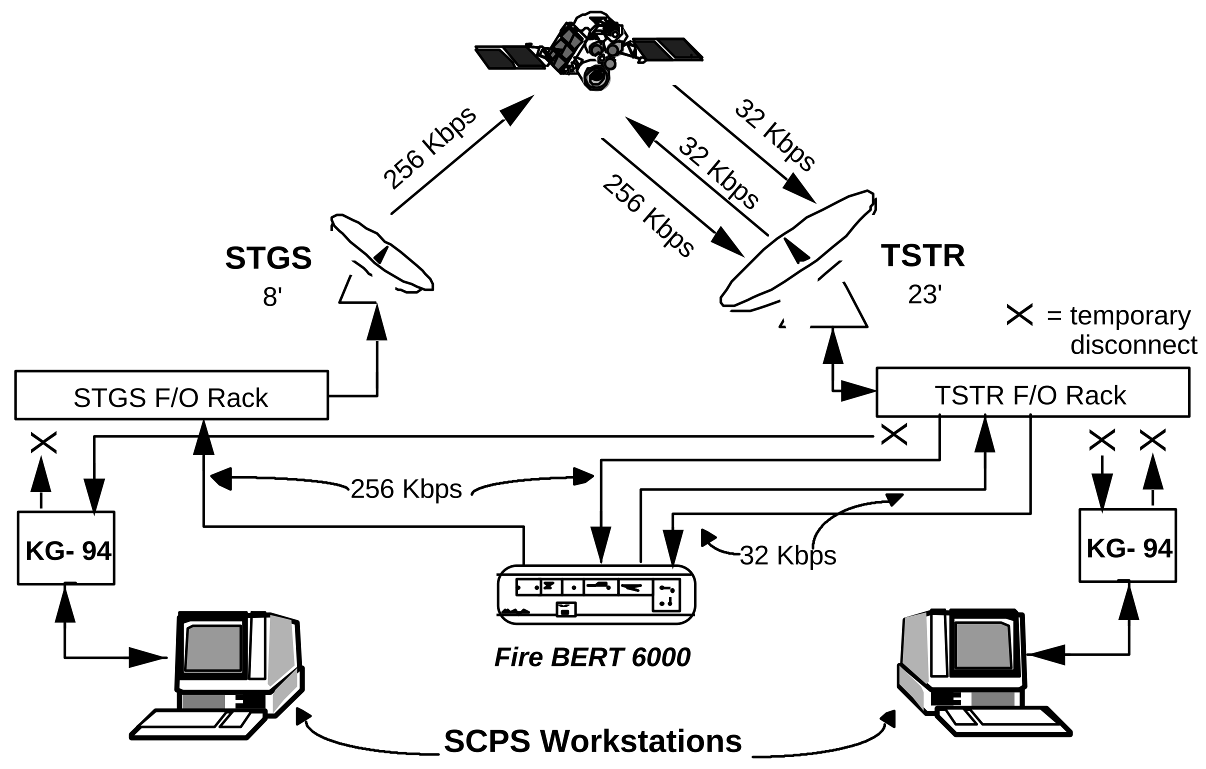


Figure 4. Bit Error Rate Test Setup

The BERTs were run by temporarily disconnecting the SCPS cryptographs from the fiber optic modem rack patch panels and connecting the BER test set. For the 256 Kbps test, the output of the test set was connected to the rack for the S-Band Transportable Ground Station (STGS) which was connected to the 8 foot antenna. The return link was connected from the rack for the Transportable Space Test & Evaluation Resource (TSTR) on the 23 foot antenna, to the input of the test set. Both the 32 Kbps forward and return links were run over the 23 foot antenna. After the tests the cryptographs were reconnected to the antennas in the same configuration as the error rate test setup. Refer to Appendix C for a complete description of the ground station configuration.

4.2.3 Cryptographic Equipment Configuration

All communications over the M-22 communications package must be encrypted. The SCPS workstations were connected to the ground station system via standard KG-94 cryptographs installed in standard HNF-81 rack mounts. Both sides of the cryptographs, i.e., the clear text and encrypted text sides were isolated from the external connections via fiber optic modems. Refer to Appendix C for the detailed illustrations for cryptograph installation and pin-out configuration.

4.2.4 Bent-Pipe Test Data Flow

Refer to Figure 5 for an illustration of the bent-pipe test data flow. For the test, data originates within one of the SCPS nodes. In these workstations, the scps_init routine generates the data, and submits it to SCPS-TP for protocol processing. The SCPS protocols transmit the data through the HSIS cards, over the RS-449 interfaces to the KG-94. The KG-94 applies the required cryptography, and forwards the data to the deployable ground station. The ground station routes the data through the satellite to the receiving ground station. The receiving ground station forwards the data to the second KG-94 for decryption. Upon successful decryption, the data is forwarded via the RS-449 interface to the second SCPS node. The SCPS protocols are terminated, and the data is forwarded to the scps_resp routine, completing the data flow. The SCPS protocols receiving the data generate acknowledgments, as appropriate, and send these back to the originating SCPS node. The path taken by the acknowledgments mirrors that taken by the original data.

4.2.5 SCPS-TP Protocol Configuration

The following table summarizes the SCPS-TP configuration information used for the bent-pipe experiments:

Table 2. SCPS-TP Configuration for Bent-Pipe Experiment

Configuration Parameter	**Bent Pipe Test Setting**
Buffer Size Send buffer size Receive buffer size	 71,929 bytes 69,881 bytes
User data per packet	1000 500 250 125
Rate control settings - varied as a function of packet size Initiator to Responder 1000 bytes of user data per packet 500 bytes of user data per packet 250 bytes of user data per packet 125 bytes of user data per packet Responder to Initiator	 (see note 1) 271 Kpbs 256 Kbps 256 Kbps 288 Kbps 32 Kbps
Congestion control	Off
Ack Frequency	Nominally 2 ACKs/RTT, with ability to increase to 16 ACKs/RTT based on circumstance (see note 2)
Selective Negative Acknowledgment	On
Window Scaling	On
TCP Timestamps	On
SCPS Header Compression	On

Note 1: Ideally, the rate control settings would reflect the channel capacity, 256 kbps. This is the setting we used for the laboratory tests. However, after beginning the bent-pipe experiments, we noticed substantially lower throughputs than we were requesting, even when SCPS-TP was not active (we were sending raw data directly over the high speed serial cards). For the 1000-byte packet and 125-byte packet tests, we increased the rate control settings until the effective rate was approximately 255 Kbps at the link layer, and then used these settings for the tests. For the 500-byte and 250-byte packets, we left the settings as they were. For a further discussion of this issue, refer to Section 6.2.1.2. Note also that the output of the high-speed serial cards, when self-clocked, generated an output data rate of 255 Kbps rather than 256 Kbps. This was the link layer data rate used for the testing.

Note 2: The software for the bent-pipe test contained several triggers for acknowledgment that no longer appear in the software. There was the normal delayed acknowledgment trigger, which would send acknowledgments twice per round trip while data was flowing. In addition, the implementation contained additional code to issue an acknowledgment whenever the receive window opened, a hole in the out-of-sequence queue closed, or whenever local buffers were beginning to fill. Much of this code has since been disabled, however, it was active for the field testing.

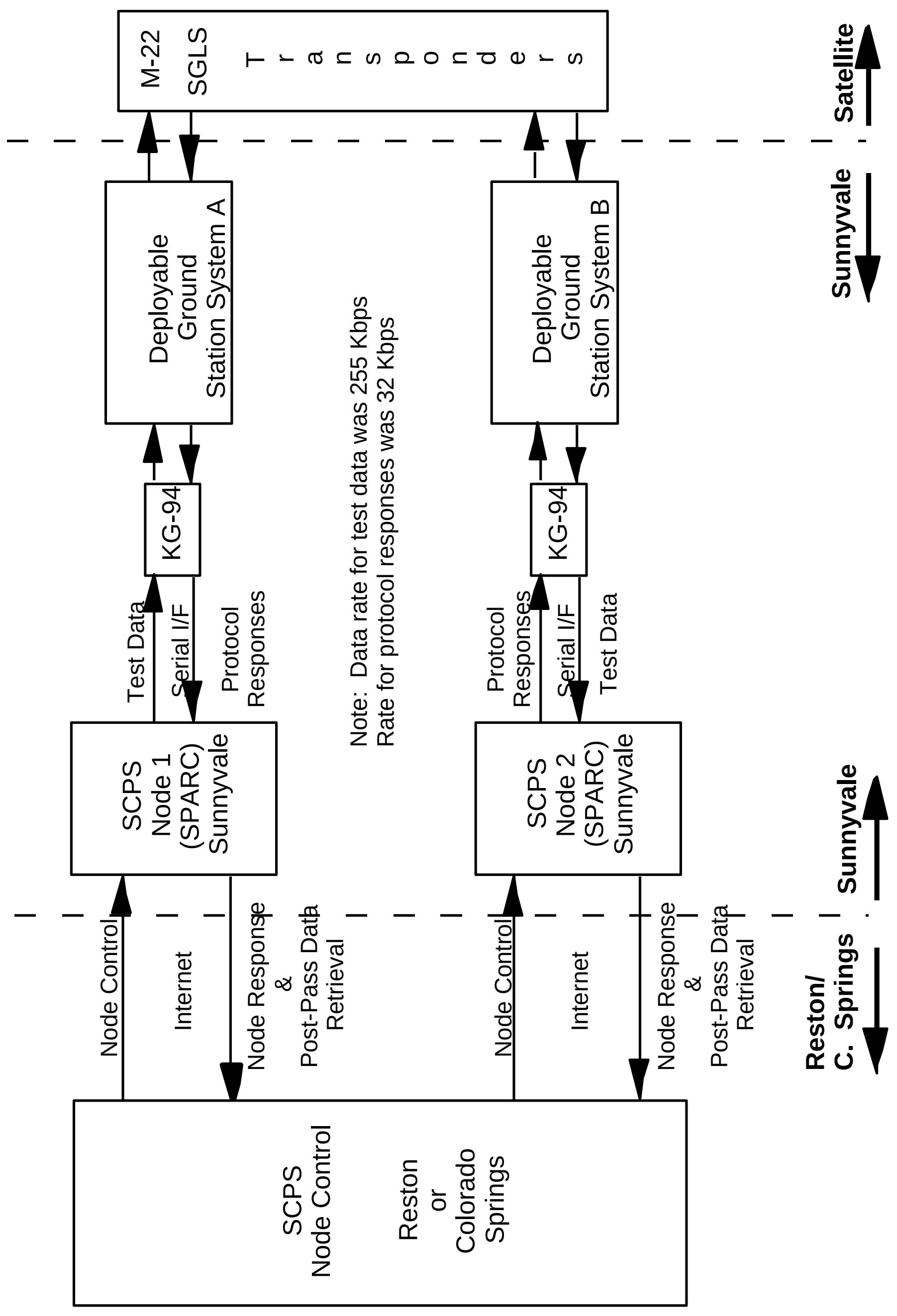
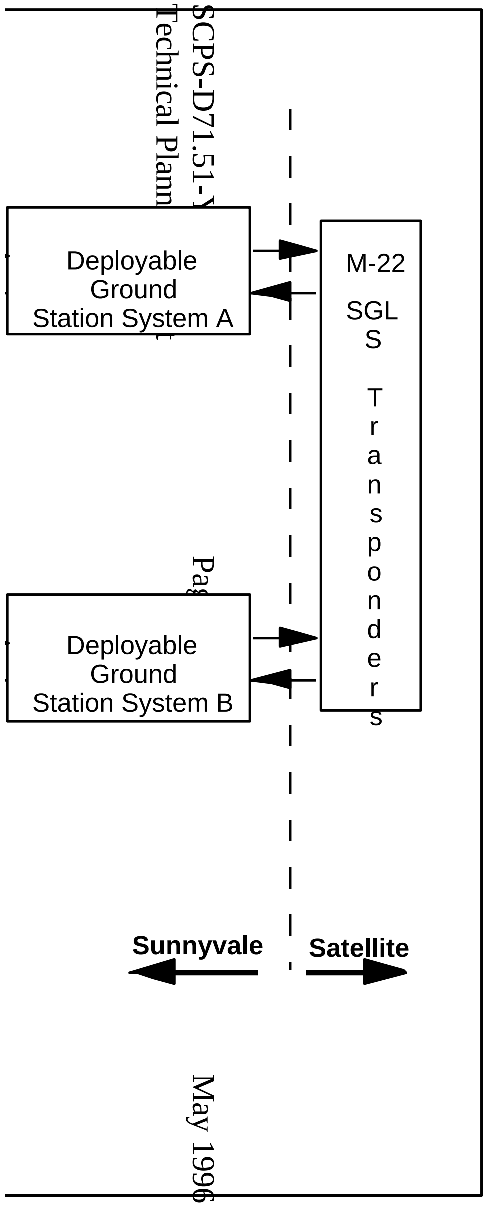

M-22
SGLS
Transponders
Satellite
Sunnyvale
Deployable Ground Station System A
Deployable Ground Station System B
KG-94
KG-94
Test Data
Serial I/F
Protocol Responses
Note: Data rate for test data was 255 Kbps
Rate for protocol responses was 32 Kbps
SCPS Node 1 (SPARC) Sunnyvale
SCPS Node 2 (SPARC) Sunnyvale
Sunnyvale
Reston/ C. Springs
Node Control
Internet
Node Response & Post-Pass Data Retrieval
SCPS Node Control
Reston or Colorado Springs

Figure 5. SCPS Bent-Pipe Experiment Data Flow


Deployable Ground Station System A
Deployable Ground Station System B
M-22
SGLS Transponders
Sunnyvale
Satellite

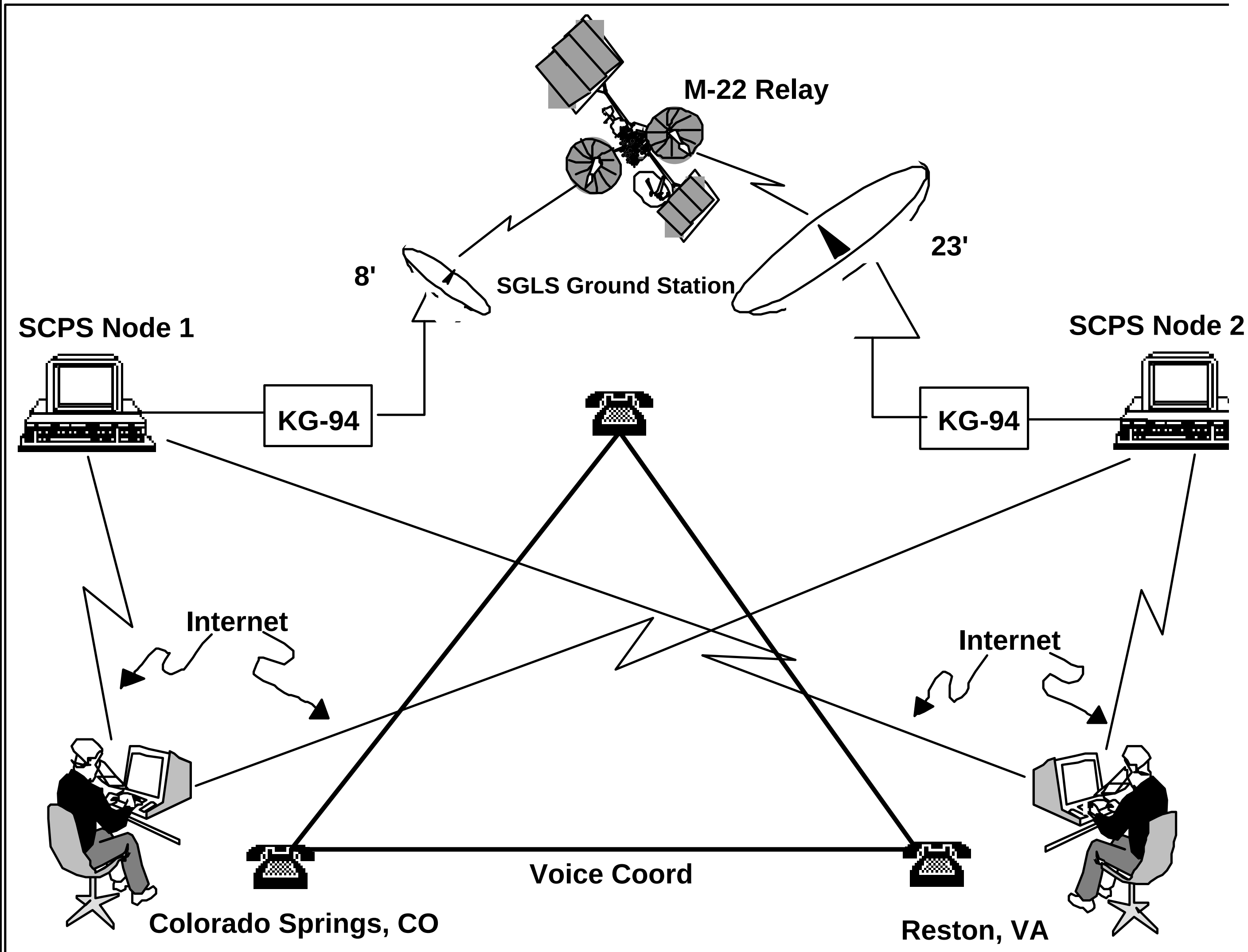


Figure 2. SCPS Bent-Pipe Experiment Configuration

5.0 EXPERIMENT METHODS

There were two types of measurements made during the experiments: measurements of the operation of the protocol under test, and measurements of the environmental characteristics during the tests. This section describes those measurements and how they were obtained.

5.1 DATA COLLECTION

The SCPS-TP implementation is instrumented to collect and present overall data about a test. This includes the total amount of data sent, the user data throughput, the elapsed time, and round trip timing information. (Refer to Appendix A.1 for the complete listing of the data.) This information is printed to the screen at the completion of each test run.

For the bent-pipe tests, we captured the screen output to a file, and labeled that file with the date and satellite supporting the test. On the first day of testing, we inadvertently corrupted some of those files, so after that day, we augmented this procedure by manually recording all of the information that was printed to the screen.

5.2 ERROR INTRODUCTION AND MEASUREMENT

There were two types of errors associated with the bent-pipe experiment: natural and induced. Natural errors were those errors that occurred on the links naturally. Most of the natural errors were caused by ground obscurations as a result of the location of the antennas and their required pointing angles to the spacecraft. For a number of reasons, the experiment was conducted using systems located at the Deployable Systems maintenance facility operated for AFMC by Loral in Sunnyvale, CA. The antennas were located near trees and buildings so that the line-of-sight was partially blocked at the beginning and end of a track, and caused a very rapid degradation in the link quality. In between these limits the links were almost always of high quality with errors on the order of zero to three per five-minute BERT. There was one track in which bursty type noise was observed during a BERT, and the operators reported that the antenna was pointing directly through a very intense thunderstorm. Whether or not the noise was caused by the storm is debatable, but the antenna was pointed in a direction in which low noise was normally encountered.

The induced, or artificial errors were required in order to be able to exercise the protocol under stress conditions for controllable periods of time. Several approaches were considered and rejected for various reasons. The method finally agreed upon was to decrease the modulation index on the 256 Kbps uplink. This solution eliminated several concerns with the other proposed methods in that it required no special equipment or modifications to the system, it affected only the signal, not the ambient noise, and it was an easy adjustment to make. It had been hoped that the adjustment could be calibrated to allow it to be set to predetermined error rates, but such was not the case. The adjustment is normally a maintenance adjustment done in a calibration mode. It is normally set for optimum bandwidth within the constraints of the SGLS signal structure. Decreasing the modulation index reduces the information carrying portion of the signal and increases the energy in the RF carrier. In terms of signal space, you reduce the distance between the signal states, making the system more susceptible to noise (phase and frequency noise in the case of a PSK or FSK system). Also, inter-symbol interference increases because the power spectrum of each desired signal state (resulting from digital data modulating the phase of frequency of the carrier) starts overlapping the spectrums of the other signal states.

In theory, it would have been possible to calibrate the modulation index adjustment control to allow it to be pre-set to the desired error rate. In practice, the precision required was not obtainable without taking the link out of service and performing time consuming repetitive adjustments and measurements during each pass. This time was not available. In the end, we were able to set the adjustment to within a predetermined range of errors. The low range was with the control set to the normal setting. The mid-range gave errors on the order of 15 to 80 per minute. The high range gave errors greater than 80 per minute, usually in the range of 150 to 250 errors per minute. Table A-4 in Appendix A shows the range of induced errors achieved during the experiment. It should be kept in mind that errors on the order of thousands per minute were encountered as the antenna beams approached the obscurations, and these data points are included if a run was able to complete in these high noise environments.

6.0 EXPERIMENT RESULTS

6.1 LABORATORY EXPERIMENT

We conducted many experiments in the laboratory with TCP and with SCPS-TP. For the TCP experiments, we were able to collect throughput information only. (The TCP implementation we used was a commercial implementation, with restricted access to the kind of internal information that we could record about SCPS-TP.) For the SCPS-TP experiments, we collected throughput information, data channel utilization information, and acknowledgment traffic information.

For each of the experiments, we transmitted six million bytes of data from the initiator to the responder over the 256 kbps channel. Acknowledgments flowed from the responder to the initiator over the low-rate channel. We varied the bit error rate from 10^{-8} to $5x10^{-6}$ for the TCP experiments, and to 10^{-5} for the SCPS-TP experiments. (The difference in the error-rate ranges results from the difference in error performance for the two protocols.)

6.1.1 User Data Throughput Results

There are two sets of throughput data presented in this section: a comparison of TCP and SCPS-TP, and a characterization SCPS-TP performance for various packet sizes and error rates.

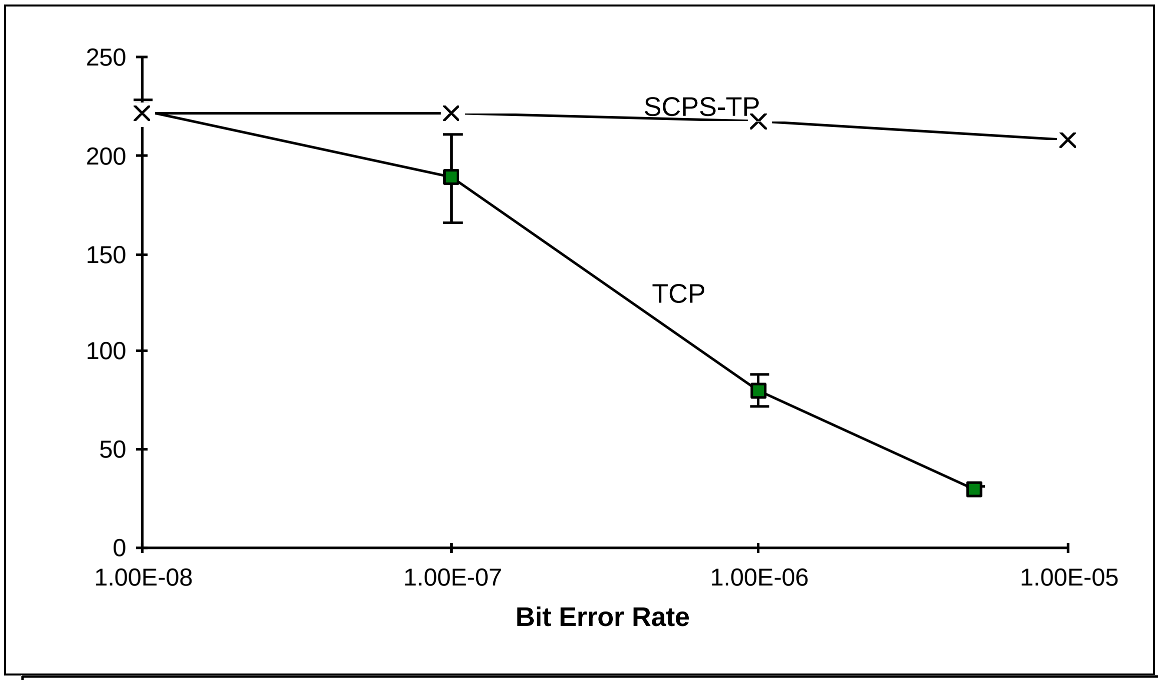
6.1.1.1 TCP comparison with SCPS-TP

Recall that the TCP maximum segment size was restricted to 512 bytes because the TCP entities were configured to believe that their peers were on different subnetworks. TCP attempts to send data in maximum-segment size packets in order to improve efficiency. As a result, we are not able to test the SunOS 4.1.3 version of TCP with other segment sizes.

We compared TCP with SCPS-TP sending 500-bytes of user data per packet. The rate control for SCPS-TP was set to 256 kbps. Figure 4 shows the throughput performance of TCP and SCPS-TP as a function of bit-error rate for 250 milliseconds one-way delay.

Figure 6 shows curves for TCP and for SCPS-TP, with data points at intervals along the curves. Each data point represents the average throughput of five samples at that bit-error rate. The error bars above and below the data points represent the 90% confidence intervals for the sampled data. (Note: the 90% confidence intervals for the SCPS-TP data are all less than 1 Kbps. As a result, the error bars for SCPS-TP are virtually indistinguishable from the line showing the average readings.)

Note that neither SCPS-TP nor TCP show throughput equal to the channel rate of 256 kbps. The throughput reported here is *user data* throughput, and doesn't take into account the overhead of protocol headers. Let us briefly consider the 10^{-8} error case for SCPS-TP: There is user data of 500 bytes per segment. We are operating SCPS-TP *without* SCPS-TP header compression, so the SCPS-TP headers are 32 bytes per packet. In addition, there is an IP packet header of another 20 bytes. In order to achieve the 222 kbps of user data throughput, SCPS-TP must use 222 kbps x ((500 + 32 + 20)/500) = 245 kbps of the channel capacity. This leaves 11 kbps (4.3%) of the channel capacity unaccounted for. A portion of this is attributable to the way in which we calculate throughput: throughput is measured from when the first data packet is sent until the last data packet is acknowledged. Therefore, there is an additional round trip time at the end of the transfer in which no data are being transferred. This reduces the unaccounted-for capacity to 3.8%. We do not have an adequate explanation for this portion of the capacity, but it appears to be constant across TCP and SCPS-TP. It is possibly traceable to the Spanner program, but we have not explored this in detail.


250
200
150
100
50
0
SCPS-TP
TCP
1.00E-08
1.00E-07
1.00E-06
1.00E-05
Bit Error Rate

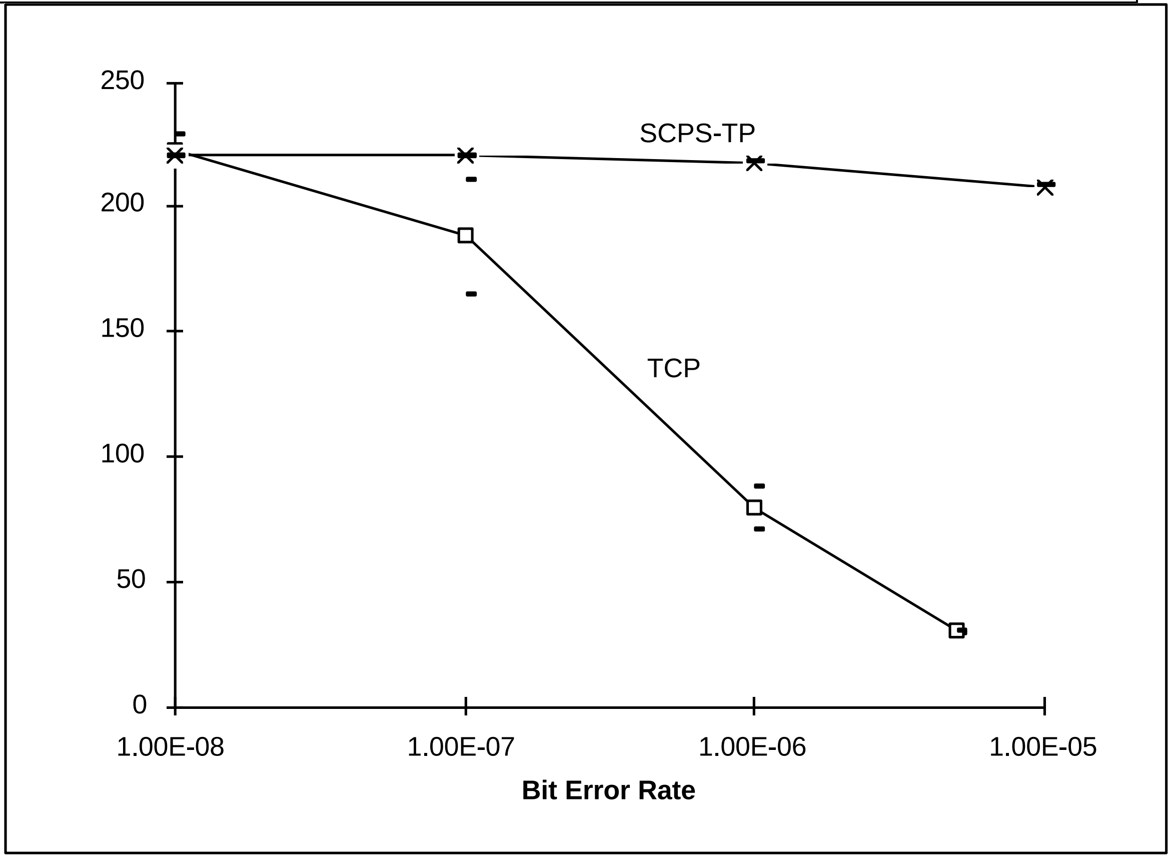

250
200
150
100
50
0
SCPS-TP
TCP
1.00E-08
1.00E-07
1.00E-06
1.00E-05
Bit Error Rate

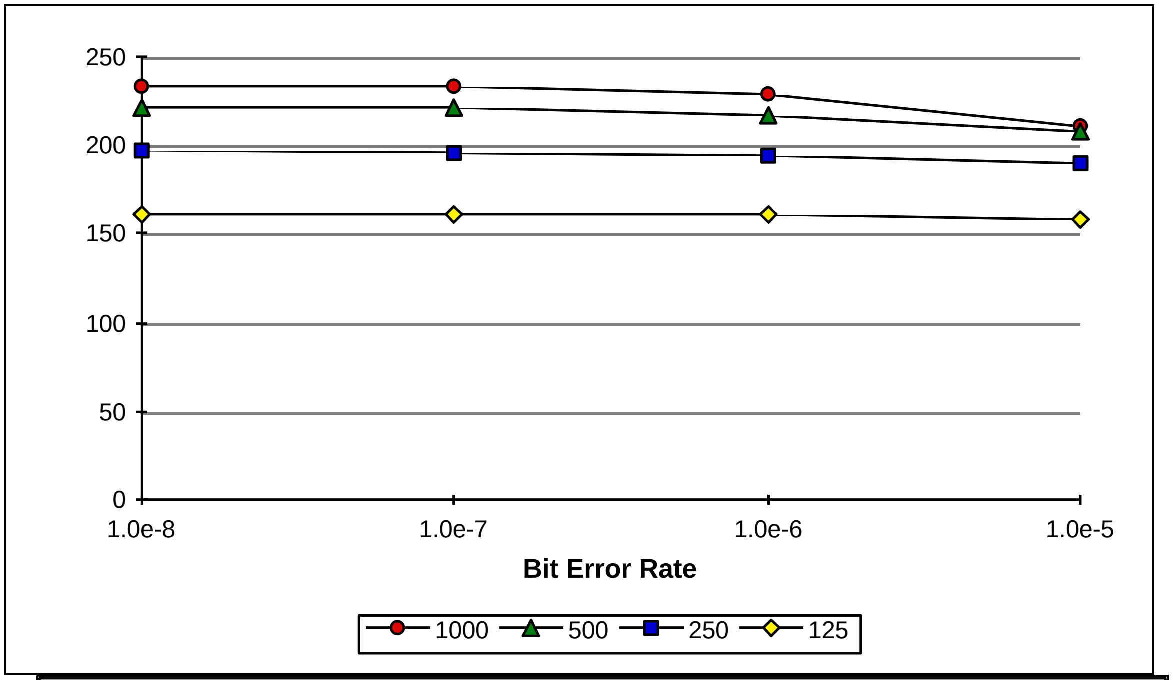
Figure 6. SCPS-TP and TCP Throughput versus Bit Error Rate

We see that at low bit error rates (e.g. 10^{-8}), the throughput for SCPS-TP and TCP is nearly identical, and that their confidence intervals overlap. However, with a bit-error rate of only 10^{-7} the throughput of TCP has fallen below that of SCPS-TP, and the 90% confidence intervals no longer overlap. TCP performance continues to drop, and has fallen to under 50 kbps at a bit-error rate of $5x10^{-6}$, while SCPS-TP throughput is still above 200 kbps.

The dramatic decrease in TCP throughput is primarily a result of TCP's assumption that all loss is attributable to congestion. In an attempt to remedy the (non-existent) congestion, TCP reduces its transmission rate and builds it back up slowly. While this response is appropriate when congestion actually does exist, it significantly degrades performance when loss is due to causes other than congestion.

6.1.1.2 Characterization of SCPS-TP

Figure 7 shows SCPS-TP throughput for a range of packet sizes. The purpose of these tests is twofold. The first goal is to characterize the throughput performance of the SCPS-TP protocols for a range of packet sizes and a range of error rates. The second goal is to determine if the behavior of the protocols in the laboratory environment is representative of the behavior in the satellite link environment.



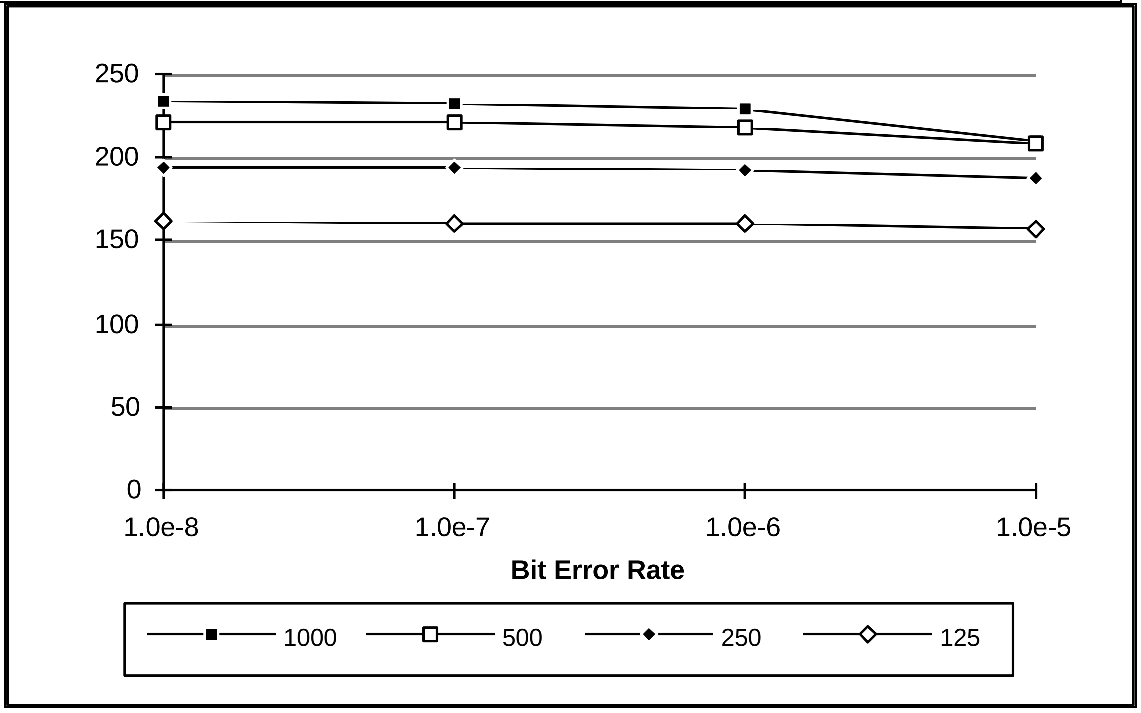


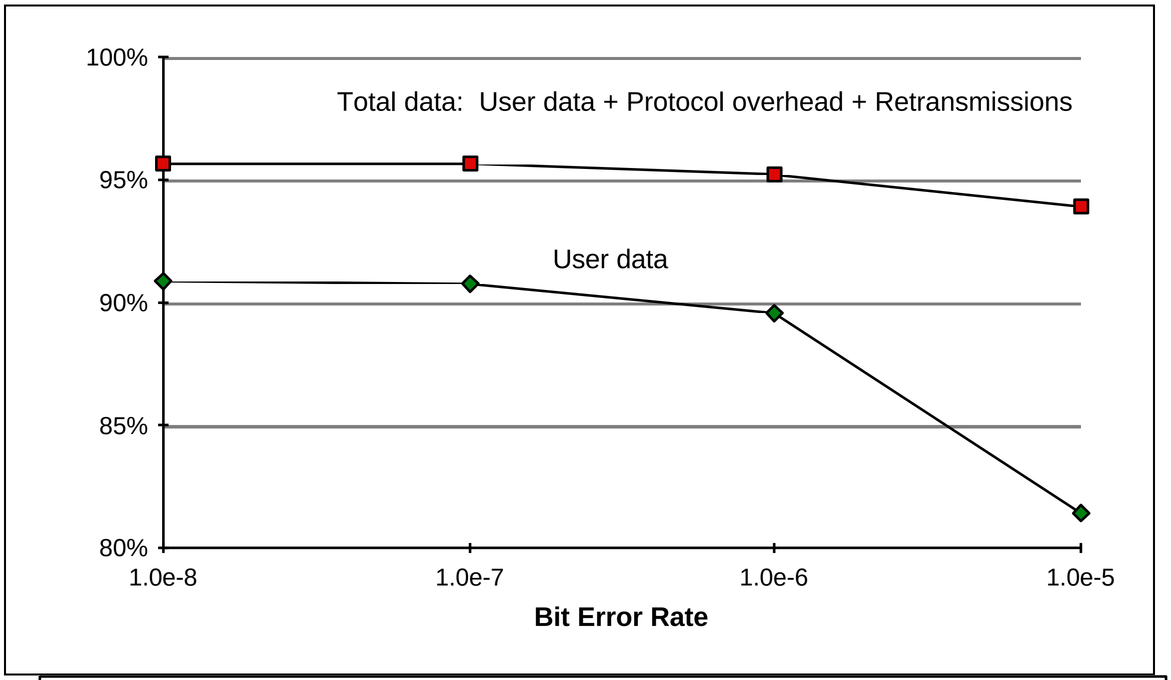
Figure 7. SCPS-TP Throughput versus Bit Error Rate for Various Packet Sizes

The legend in Figure 7 identifies the amount of user data in each packet transmitted. The tests show that SCPS-TP throughput is relatively unaffected by bit-errors for bit error rates better than 10^{-5}. As the bit error rate increases from 10^{-8} to 10^{-5}, the drop in throughput for 1000-byte packets is 10.3%, while the throughput loss for the smaller packet sizes is less than 5%.

6.1.2 Data Channel Utilization

An examination of data channel utilization provides insight into protocol performance and the optimality of the protocol configuration. Refer to Figure 8, below. The figure shows the data channel utilization as a percent of total channel capacity. (In other words, 100% channel capacity equals 256 kbps.) The upper line on the graph shows the utilization of the 256 kbps channel when transmitting 1000-byte packets. The lower line on the graph shows the percent of that channel capacity devoted to the (initial) transmission of user data. (The upper line accounts for all protocol overhead and all retransmissions of user data.) When properly configured, the channel utilization will be relatively constant for the full range of error rates.

Note that overall data channel utilization drops gradually as the bit error rate increases. This indicates that the protocol is not continually transmitting (either new data or retransmissions) as the error rate increases. This fall-off, which represents only 1.5% of the channel capacity, is a subject for further investigation at a later date. Note also that the percent of channel capacity devoted to (initial) transmission of user data drops by 9%. At an error rate of 10^{-8}, the difference in overall utilization and the user data utilization represents the overhead of protocol headers. The 9% fall in user data utilization represents channel capacity that is devoted to retransmissions, “crowding out” the transmission of new data.



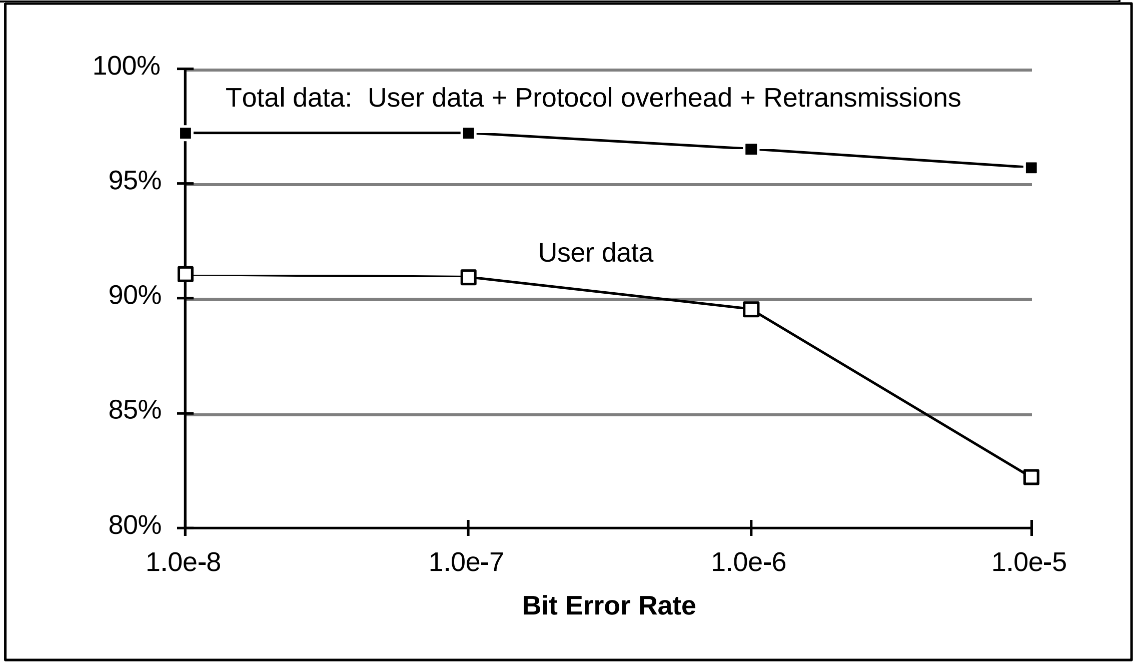
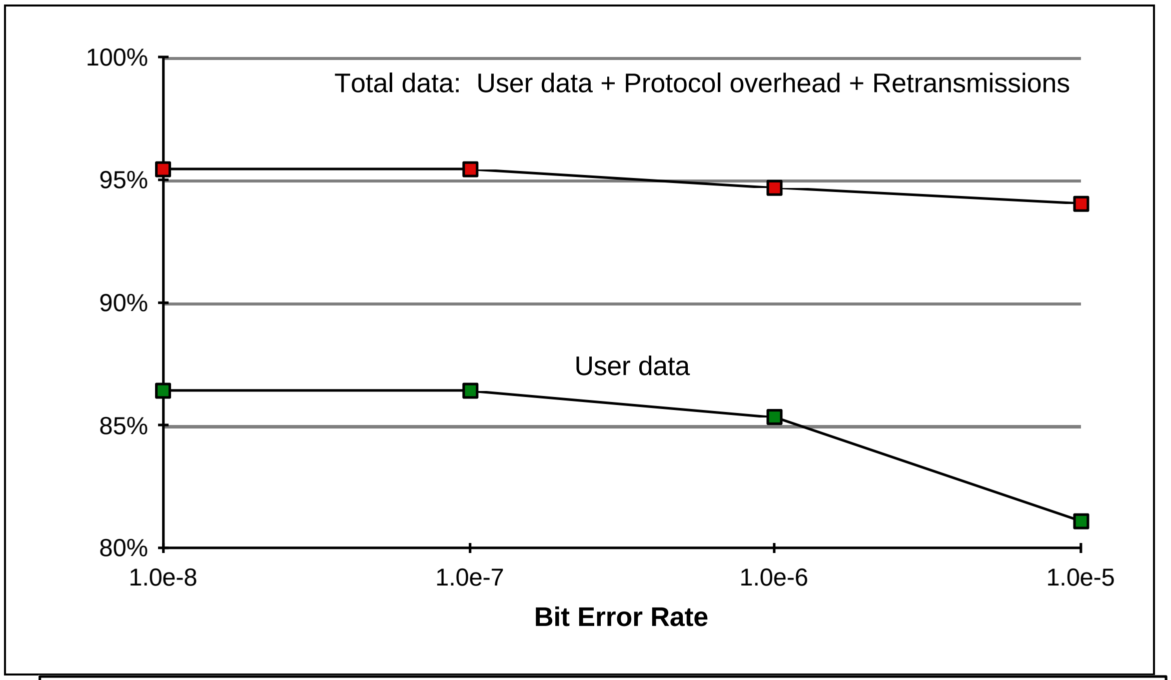


Figure 8. User Data Versus Total Data Use of Data Channel for 1000-byte Packets

Figure 9 shows the same type of graph, but for 500-byte packets. There are two principle differences between this graph and the graph in Figure 8. First, the difference between the Total Data and User Data line is 9% at an error rate of 10^{-8}. This is greater than the 5% difference seen in Figure 8, and is consistent with the fact that there is half as much user data per packet (that is, there are twice as many headers transmitted for the same overall volume of data). The second difference is that between the error rates of 10^{-8} and 10^{-5}, the reduction in channel capacity devoted to User Data falls by 5%. This is less than the drop for 1000-byte packets, and illustrates the fact that fewer of the shorter packets are being lost, and therefore do not require retransmission.



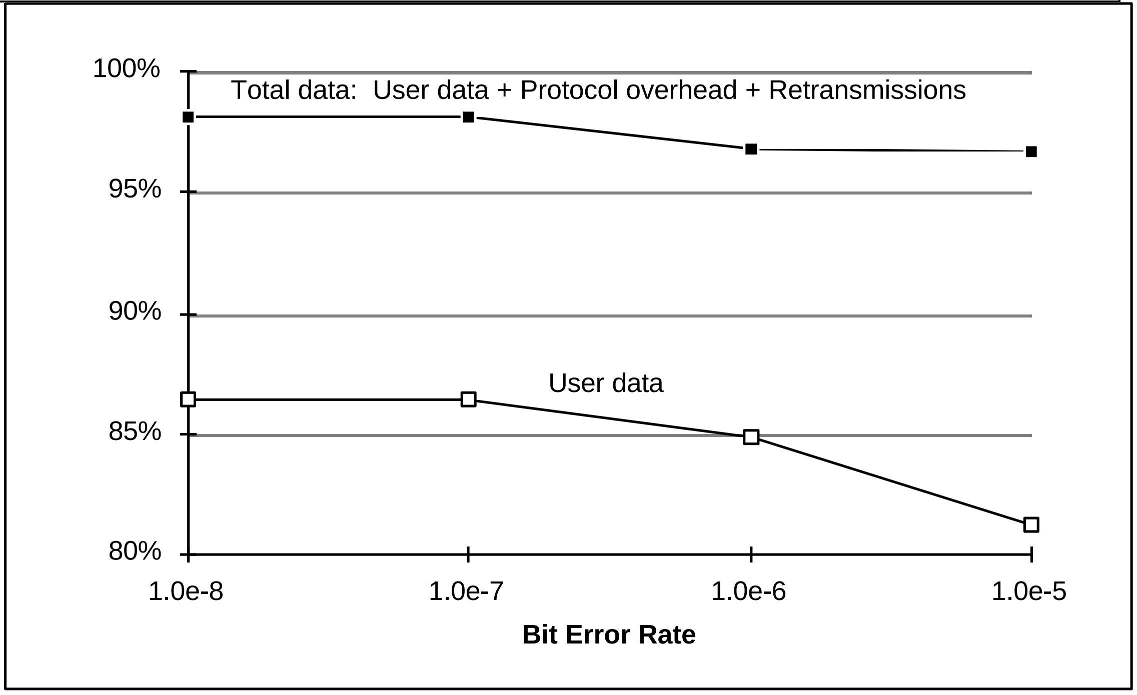


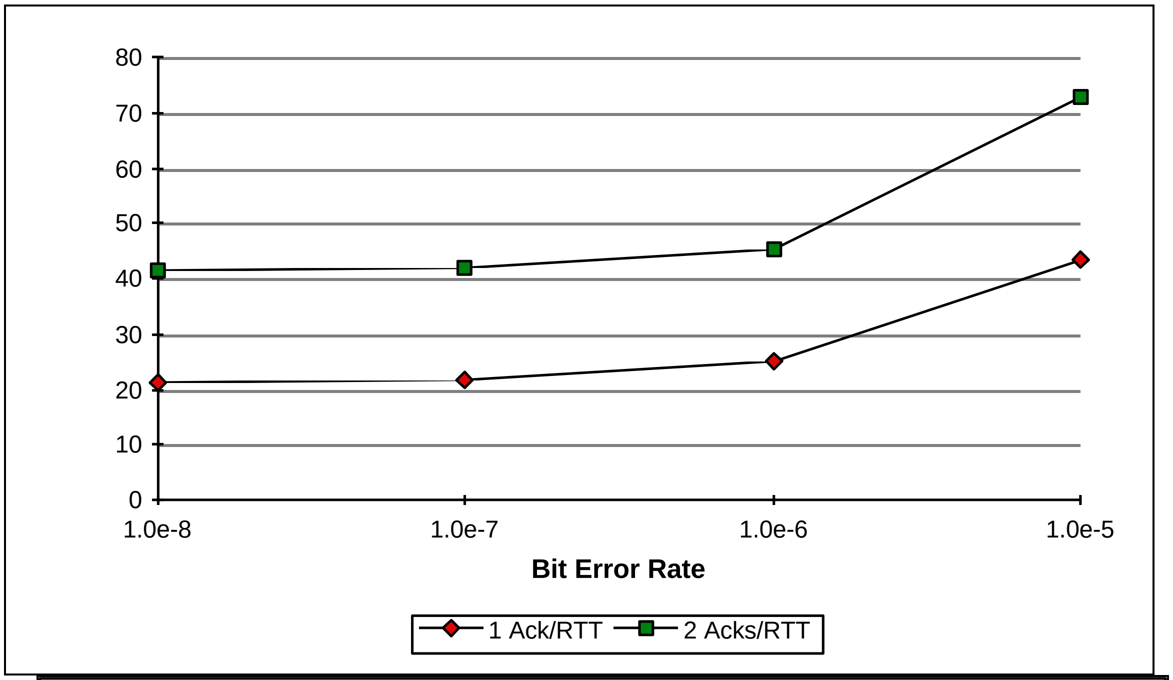
Figure 9. User Data Versus Total Data Use of Data Channel for 500-byte Packets

6.1.3 Acknowledgment Traffic

In many satellite communication systems, the data rates in each direction are not identical. There is frequently a significant disparity in data rates. When data is being transferred over the high rate link, one must be conscious of the load that this imposes on the acknowledgment channel. TCP has rules that require an acknowledgment for at least every other data packet. With SCPS-TP, we have revised the acknowledgment strategy in an attempt to reduce the loading on the acknowledgment channel. For the prototype version of SCPS-TP, we have the ability to configure the rate at which acknowledgments are sent.

Using this configurability, we can examine the effect on acknowledgment channel use and on data channel throughput. For the operational version of SCPS-TP, we intend to identify the means to allow SCPS-TP to adapt its acknowledgment rate to an appropriate setting in response to changing communication conditions.

Figure 10 shows the effect of two different acknowledgment rates on the use of the acknowledgment channel. (The data channel is carrying 1000-byte packets.) The rates are one acknowledgment for every round trip time (i.e. one acknowledgment per half second), and two acknowledgments for every round trip time.



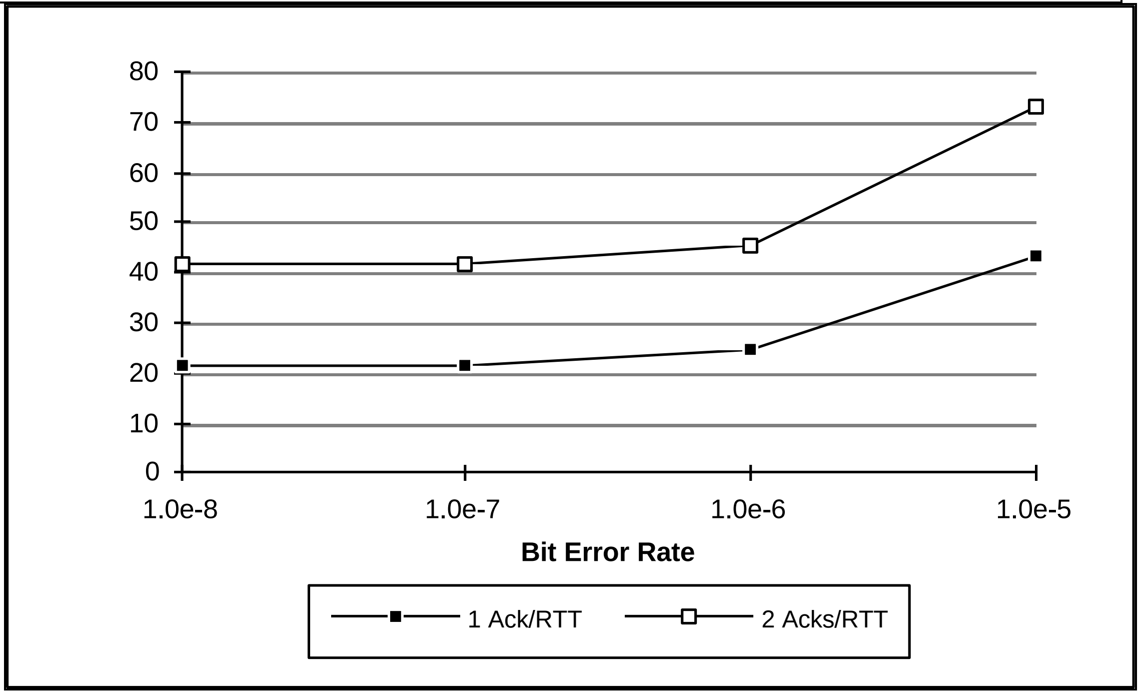
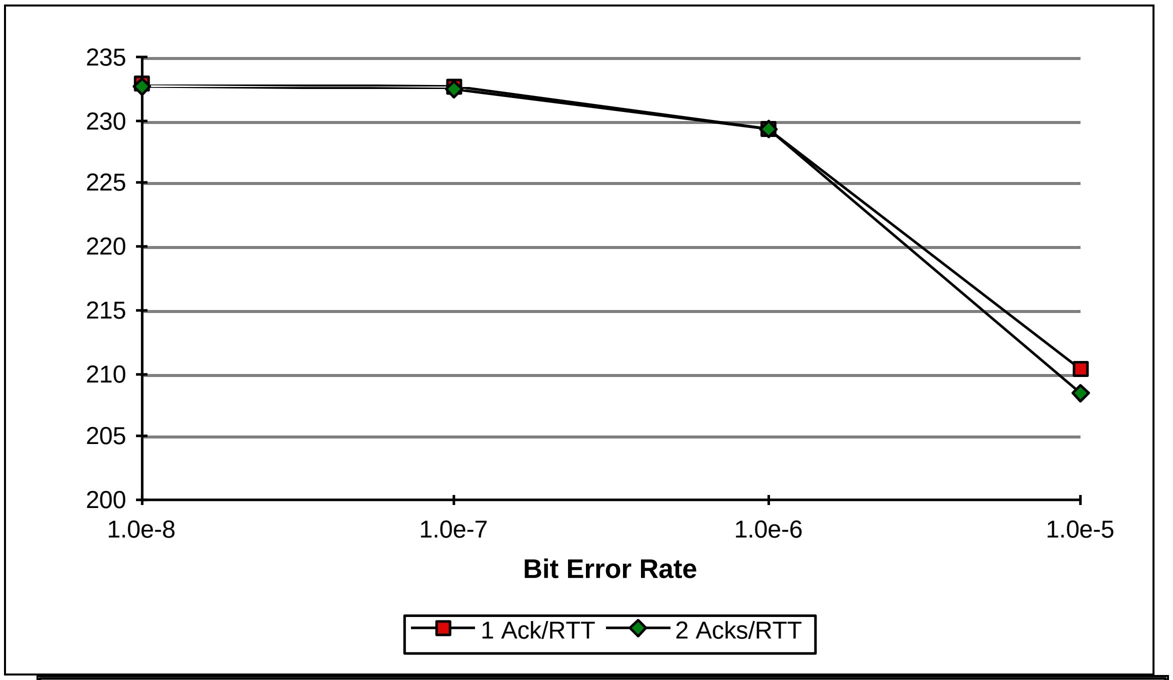


Figure 10. Effect of Acknowledgment Rate on Acknowledgment Data Sent

The effect of these acknowledgment rates on throughput is shown in Figure 11. Although not shown on the graph, the 90% confidence interval for the 1 Ack/RTT, 10^{-5} bit error rate point is 206 Kbps to 210 Kbps. The 90% confidence interval for the 2 Acks/RTT, 10^{-5} bit error rate point is 209 Kbps to 211 Kbps. Since the confidence intervals overlap, there is no statistically significant difference in throughput as a result of varying the acknowledgment rate. When we examined the throughputs for 500-, 250-, and 125-byte packets, there was no statistically significant difference in throughput resulting from a change in acknowledgment rate. Therefore, a single acknowledgment per round trip time appears to be adequate *in this configuration.* We are not convinced that a single acknowledgment per round trip time is appropriate in all situations, and will continue to investigate this issue.



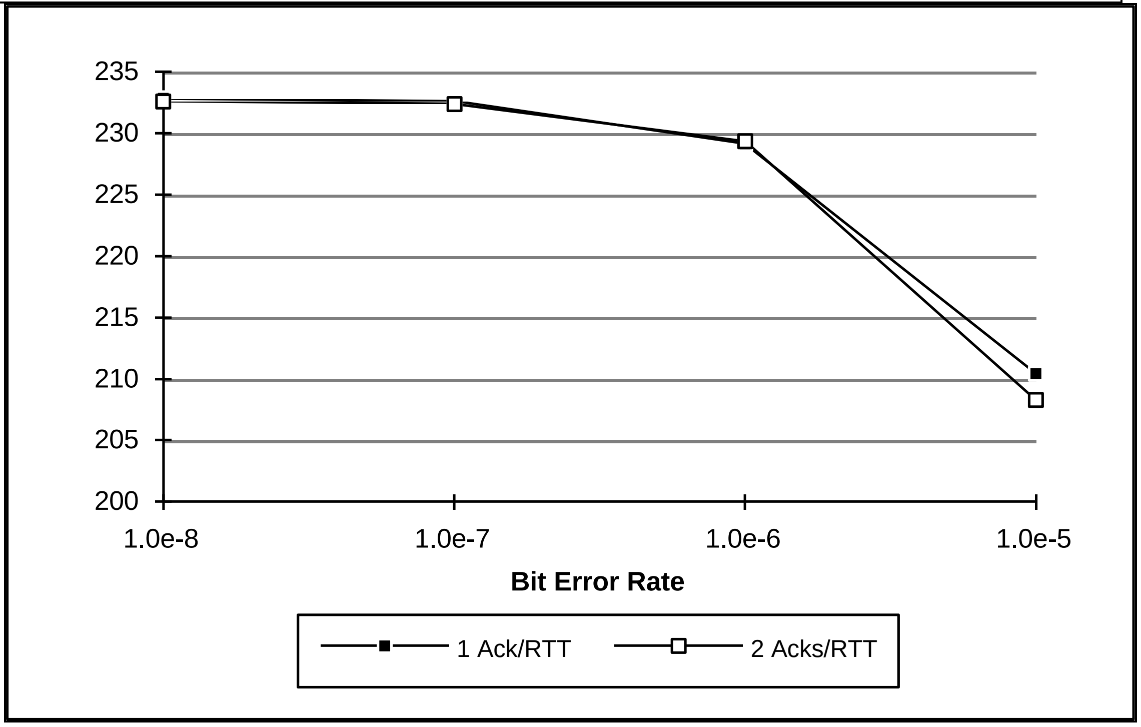


Figure 11. Effect of Acknowledgment Rate on User Data Throughput

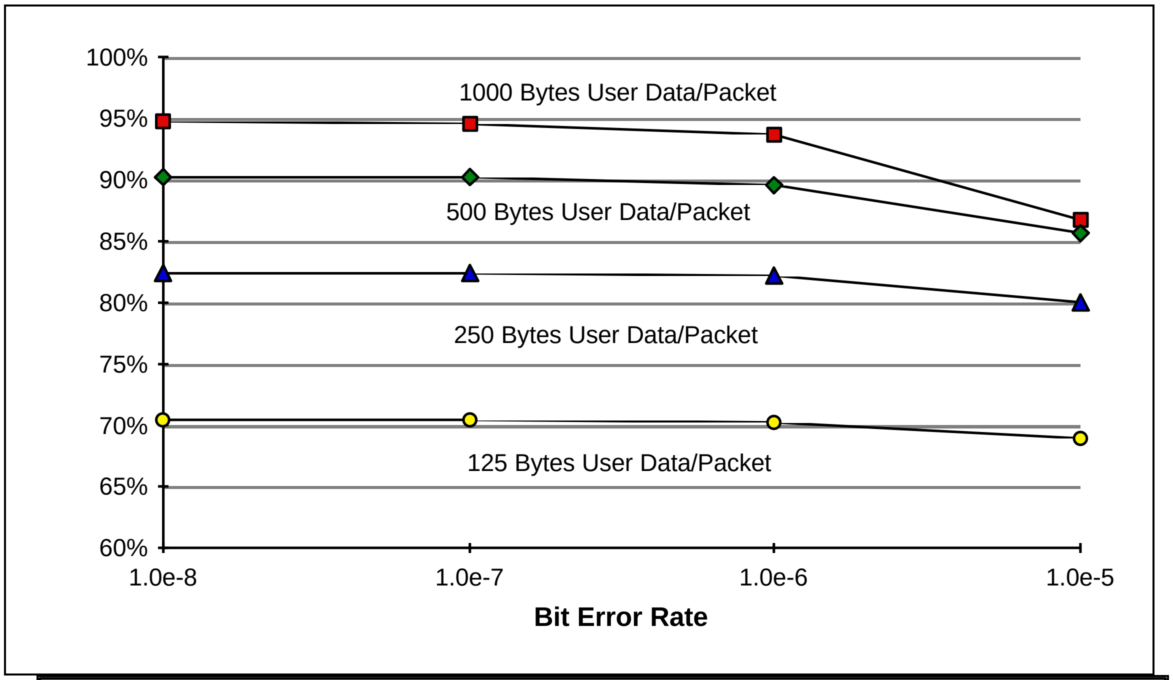
By examining this type of data for a broad range of packet sizes and error rates, we hope to be able to either statically configure SCPS-TP with an optimal rate or develop an adaptive algorithm to optimize the acknowledgment rate.

6.1.4 Bit Efficiency

Bit efficiency measures the protocol data overhead involved in transmitting a user's data. It is the fraction of the total number of bytes transmitted that are the user's data. If there were no protocol overhead nor retransmissions, the bit efficiency for a data transfer would be 1.0. However, protocol headers, acknowledgments, and retransmissions reduce that value from 1 to something less. Packet size has an effect on bit-efficiency. Protocol headers are applied on a per-packet basis; the longer the packets, the fewer instances of protocol header overhead (for a given volume of data). Therefore, large packets provide greater bit-efficiency, at least at low error rates. However, large packets are more likely to be affected by bit-errors. These packets must be retransmitted, reducing bit-efficiency.

Figure 12 shows the bit-efficiency of SCPS TP over IP. There was a goal, but no firm requirement, that SCPS TP over SCPS NP impose no more than 10% bit-overhead *in the no-error case*. We can see that this goal is met for 1000-byte and 500-byte packet. Additionally, the goal is still met for 1000-byte and 500-byte packets at bit error rates better than approximately 10^{-6}. (Recall that SCPS-TP header compression is not in use. We shall see the difference that it makes in Section 6.2.4.) The SCPS NP header size is variable, and the effect of replacing IP with SCPS NP will also vary. SCPS-NP headers can be as small as 5 bytes, which would result in a 15-byte-per-packet savings over IP. This represents only 1.4% of one of the 1052-byte (1000 bytes of user data) packets used in this experiment, but it represents 8.4% of a 177-byte (125 bytes of user data) packet, and 29% of the size of a 52-byte acknowledgment.

Note that as retransmissions increase, bit-efficiency falls. This is because retransmissions are considered protocol overhead. As the bit-error rate increases, the probability of packets being corrupted increases. With the error distribution used in the laboratory tests (i.e., a uniform distribution), larger packets have a higher probability of being corrupted than small packets, and retransmissions of large packets have a greater effect on bit-efficiency than do retransmissions of small packets. Therefore, as the bit-error rate increases, we would expect that the bit efficiency for the larger packet sizes would eventually cross that of the smaller packet sizes. In Figure 12 we see that this phenomenon is happening relative to the 1000-byte packets and the 500-byte packets, but the crossover is to the right of 10^{-5}.



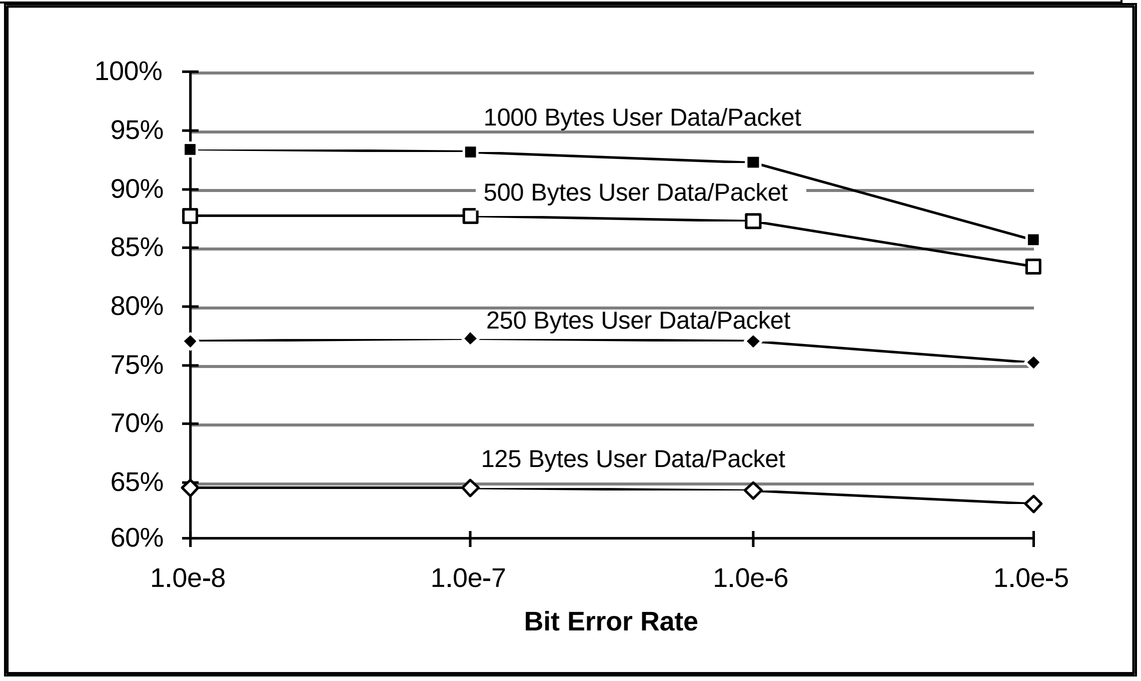
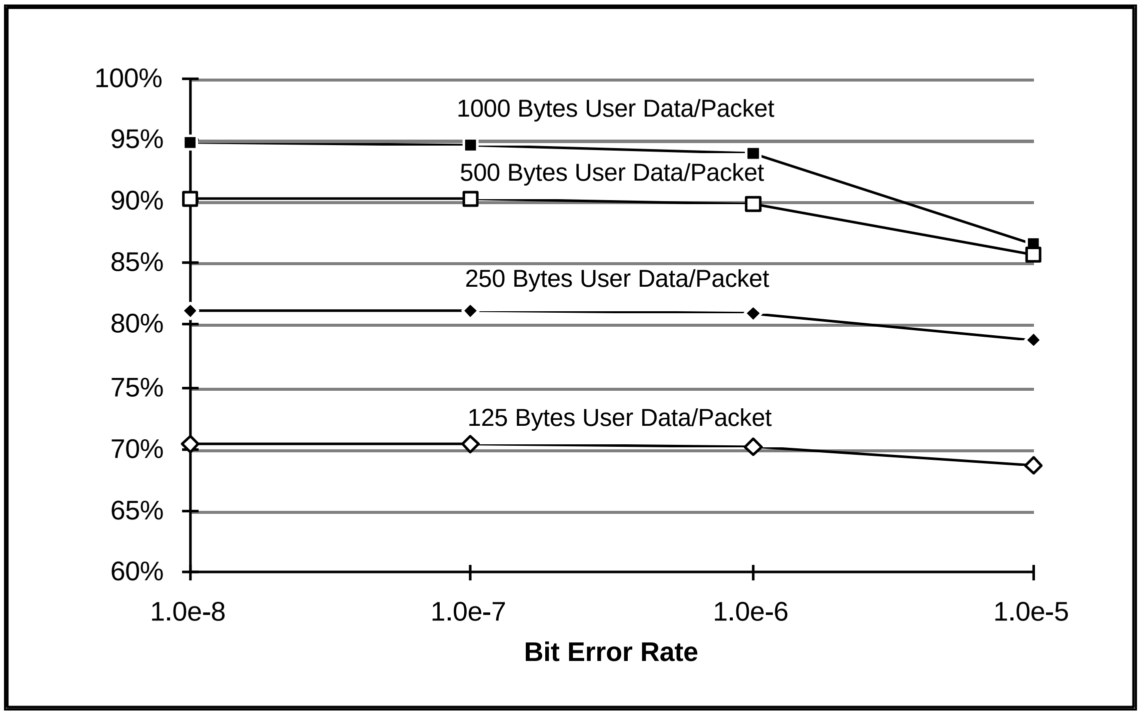


Figure 12. Bit Efficiency of SCPS-TP Over IP


100%
95%
90%
85%
80%
75%
70%
65%
60%
1000 Bytes User Data/Packet
500 Bytes User Data/Packet
250 Bytes User Data/Packet
125 Bytes User Data/Packet
1.0e-8
1.0e-7
1.0e-6
1.0e-5
Bit Error Rate

6.2 BENT-PIPE EXPERIMENT

6.2.1 User Data Throughput Results

This section presents the bent-pipe experiment throughput results for each packet size. Each packet size is presented on a separate graph, with the laboratory throughput information plotted for reference. We summarize the results, present a discussion of the results when appropriate, and then draw conclusions. Readers not interested in a detailed discussion of each graph may wish to read the introductory paragraph of each subsection, then skip to the conclusions at the end of each subsection.

6.2.1.1 Throughput analysis of 1000-byte packet tests

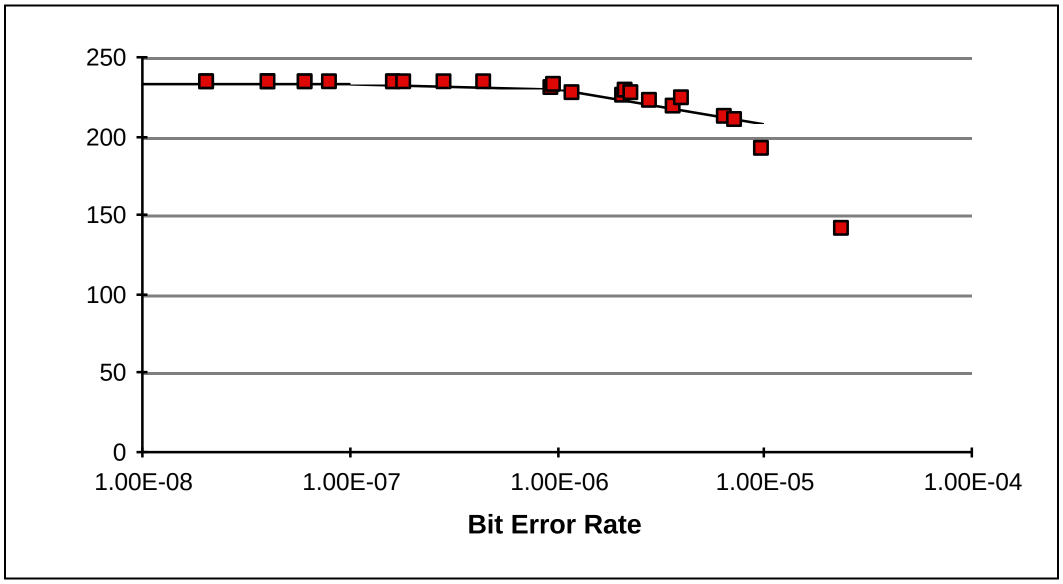


Figure 13. SCPS-TP Throughput Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figure 13 shows the throughput of SCPS-TP versus bit-error rate for the laboratory data (shown as a thin, solid line) and the bent pipe test data. The laboratory data is shown as the mean of five runs at each of the following four bit-error rates: 10^{-8}, 10^{-7}, 10^{-6}, and 10^{-5}. The bent-pipe test data is shown as individual samples, with throughput as measured, and bit error rate estimated as described in Section 6.3. For this graph, each SCPS-TP segment (packet) contained 1000 bytes of user data. We see that the throughput performance of SCPS-TP is almost flat for bit-error rates better than 5×10^{-7}. Between 5×10^{-7} and 10^{-5}, we see a 18.5% reduction in throughput, and a 32.5% reduction in throughput (from the 5×10^{-7} value) at a BER of 2.3×10^{-5}.

Q: Why is the throughput of the bent-pipe test data higher that of the laboratory data (at BERs better than 10^{-5})?

A: The laboratory data was taken with SCPS-TP header compression turned off, while header compression was on for the bent-pipe test. The use of header compression for the bent-pipe test will result in higher throughput. For 1000 bytes of user data per packet (the packet size shown), the difference in header size is approximately 1.5%. At 255 Kbps, the difference in throughput of 1.5% should be approximately 3.9 Kbps. We observe an actual difference of approximately 2 Kbps. The discrepancy was due to inaccuracy in the rate control mechanism used for the bent-pipe test, which is described in the following section.

Q: Why is the throughput at 10^{-5} lower for the bent-pipe test than for the laboratory data?

A: The bent-pipe test data point in question has a throughput of 192 Kbps, while the mean throughput of the laboratory data was 208 Kbps. The bent-pipe test sample was taken on 12/13/95 at 20:33. An examination of the responder log for that run revealed that there was a memory buffer problem that reduced the amount of memory available to the connection. Examination of the initiator log showed that the initiating SCPS-TP entity went into the Persist state nine times, indicating that responder was out of memory and that this condition was affecting the ability of the initiator to maintain its data transmission rate.

Q: Is the data point (141 Kbps) at the rightmost BER reasonable?

A: Probably not - the same memory problem that affected the previous data point was probably affecting the 141 Kbps run also. At the time the run was started (12/12/95, 17:52), the link quality was degrading rapidly (a BERT at 17:45 indicated a BER of 4.4×10^{-6}, while a BERT at 18:08 indicated a BER of 9.15×10^{-4}). We do not have the initiator or responder logs for this run to determine with certainty whether a memory problem occurred, however, the amount of acknowledgment data sent is consistent with that situation.

Conclusions:

The throughput for 1000-byte packets is consistent between the laboratory data and the bent-pipe data. The bent-pipe data shows that a relatively high portion (234 Kbps/255 Kbps, or 92%) of the channel capacity is devoted to user data at BERs less than or equal to 10^{-6}. The protocol shows consistent throughput through 5×10^{-6}, then rolls off at 10^{-5}. We believe that the roll-off at 10^{-5} is too steep, due to memory problems that have since been corrected, and if repeated, would more closely resemble the laboratory data.

6.2.1.2 Throughput analysis of 500-byte packet tests

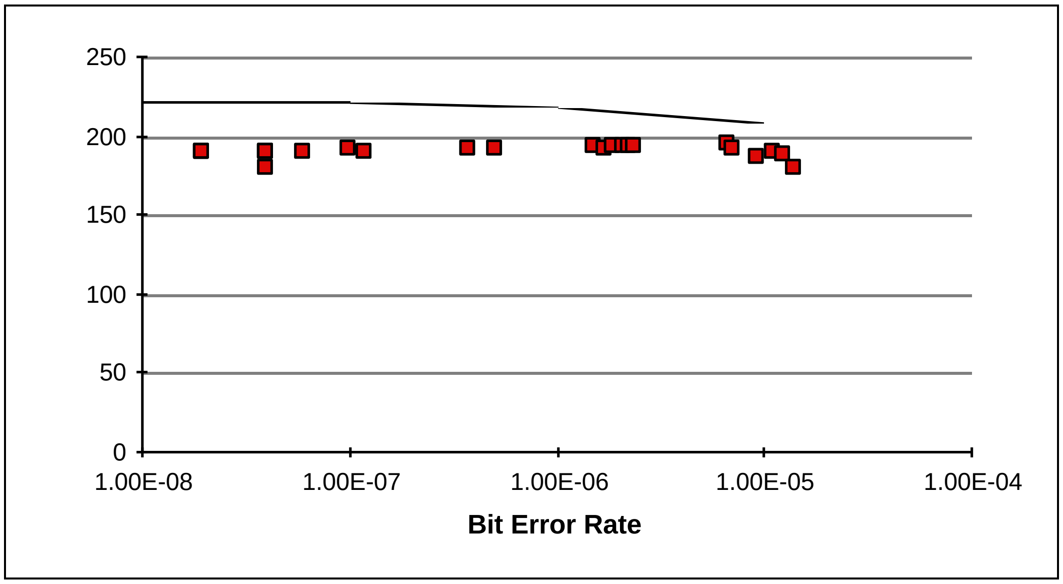


Figure 14. SCPS-TP Throughput Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figure 14 shows throughput of SCPS-TP versus bit-error rate for the laboratory data and for the bent-pipe test data. The difference between this graph and Figure 13 is that each SCPS-TP segment shown in Figure 14 carried 500 bytes of user data, rather than 1000 bytes of user data. We see that the throughput for the experiment data is uniformly lower than that of the laboratory data. We also note that the throughput is relatively flat to an error rate of almost 10^{-5}. The throughput at a BER of 1.37×10^{-5} is approximately 94% of the throughput experienced at error rates better than 10^{-7}.

Q: Why is the throughput of the bent-pipe test data lower than that of the laboratory data?

A: The rate control settings were different. We realized after we had taken the bent-pipe experiment data that our rate control settings for the experiment were flawed, because of a previously unknown relationship between these settings and the purported accuracy of the Sun system clock. The high-speed serial cards that we used to interface with the satellite terminals did not perform as expected, and we were forced to make field-adjustments of the rate control parameters to compensate. The rate control is set using two parameters: a time increment, and the amount of data (in bytes) to send per time increment. The problem we experienced had to do with setting the time increment. The Sun workstations have a system clock, which may be read by user processes for timing purposes. This clock "ticks" at a relatively slow rate (60 Hz). The system clock *appears* to be accurate to the microsecond, however (the least-significant digit of the microseconds value changes). Accurate timing using the system clock can only be done if the timing intervals are multiples of the system clock period (which, for a 60 Hz clock, is 16.667 milliseconds). If a timing interval is not a multiple of the system clock period, the actual interval will never be shorter than that requested, and may be up to a full system clock period longer than requested. For the bent-pipe experiments, the rate control intervals were not a multiple of the clock period. Not realizing the significance of this at the time, we compensated for the low observed data rates by increasing the bytes-per-interval settings for the 1000 and 125 byte runs. We did not adjust the bytes-per-interval settings for the 500 and 250 byte runs. By empirically "tuning" the 1000 and 125 byte values, we were able to achieve the desired data rates.

We were able to reproduce the rate control problem in the laboratory, and reproduced the experiment data rates using the rate control settings that we used for the experiment. When we ensured that the rate control interval was an even multiple of the system clock period, our rate control accuracy improved significantly. With the current rate control algorithm, our measured data rate has improved to within 3% of the requested data rate. However, the laboratory data was taken before this problem was identified and repaired. The laboratory data is affected by this problem, but to a much lesser degree than the experiment data is.

Q: Why are the laboratory data and experiment data throughputs lower for 500-byte packets than for 1000-byte packets?

A: The experiment data throughput is lower for the reasons described immediately above. However, the laboratory data throughput is lower for 500-byte packets because there is more protocol header overhead associated with a given volume of data with the smaller packet sizes. The laboratory data was taken using IP as the network layer protocol and uncompressed TCP with 12 bytes of timestamp information per packet. This resulted in a 52-byte header per packet, regardless of the amount of user data it carries. This overhead would lead us to expect that the 500-byte packets would show approximately 95% of the throughput of the 1000-byte packets (in the absence of errors). (For a quick example, consider a case in which we wish to send 1000 bytes of user data. For 1000-byte packets, the total data transmitted will be 1052 bytes, while for 500-byte packets the total data will be 1104 bytes. The throughput relationship will be 1000/1052 = n * 1000/1104. Solving this equation shows than n = 0.953.) If we examine the laboratory data, we see that the throughput at a 10^{-8} error rate for 500-byte packets is 221.2 Kbps, which is 95.1% of the throughput for 1000-byte packets, 232.7 Kbps.

Q: Why is the highest throughput of the experiment data *not* at the lowest bit-error rate?

A: We do not know with certainty. The effect is minor: the throughput at a BER of $6.6x10^{-6}$ is approximately 3% higher than that of the lowest BER measured. We suspect that the increase is caused by a bug that was in the software that permitted violation of the rate control settings when retransmissions occurred (this violation affected the transmission of new data as well as of retransmissions). This is combined with the fact, mentioned above, that we are significantly underutilizing the link capacity due to the particular rate control settings used for the run. (Were the link being utilized at its full capacity, this effect would not have occurred.) The increase in throughput is truly a bug, rather than a feature, because the rate control logic is there, in part, to permit sharing link bandwidth with other applications. If the protocol violates the bandwidth allocation permitted it through the rate control settings, it will probably be at the expense of other users of the link. We attempted to recreate this effect in the laboratory, but have not yet been able to do so. The current implementation of the software does not exhibit this behavior.

Conclusions:

The throughput for the 500-byte packet size experiment data was lower than expected due to a problem with the rate control parameters. This problem has been analyzed and corrected.

Throughput at the highest BER measured ($1.37x10^{-5}$) is relatively high: 94% of the throughput at the lowest BER measured, for both the laboratory and experiment data.

We noticed a minor increase in throughput with increasing bit-error rate. We do not fully understand this phenomenon, but suspect that it is the result of a violation of the rate control settings combined with the fact that we were underutilizing the link. We are continuing to investigate this issue.

6.2.1.3 Throughput analysis of 250-byte packet tests

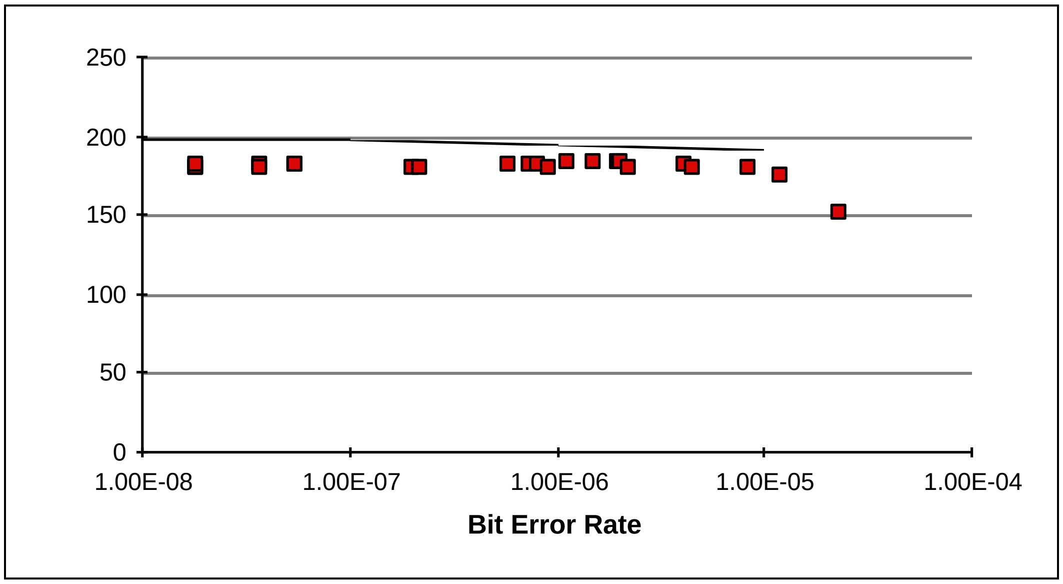


Figure 15. SCPS-TP Throughput Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figure 15 shows throughput of SCPS-TP versus bit-error rate for the laboratory data and for the bent-pipe test data with 250-byte packets. We see that, as with the 500-byte packet data, the throughput for the experiment data is uniformly lower than that of the laboratory data. We also note that the throughput is relatively flat to an error rate of almost 10^{-5}. The throughput at a BER of 1.2×10^{-5} is approximately 96% of the throughput experienced at error rates better than 10^{-7}, and the throughput at a BER of 2.3×10^{-5} is still 84.3% of that experienced at error rates better than 10^{-7}.

Q: The throughput of the experiment data is lower than that of the laboratory data. Is this due to the same rate control problem described in the throughput analysis of 500-byte packet tests?

A: Yes. Refer to the discussion in the 500-byte packet throughput analysis section, above.

Q: Does this data show the same relationship between throughput and bit-error rate that the 500-byte data does?

A: Yes, but to a lesser degree. The phenomenon, described in the 500-byte packet throughput analysis section, yields slightly higher throughputs in the 10^{-6} BER range than are seen at lower BERs. As mentioned previously, we are continuing to investigate this issue, but believe that it is caused by a combination of a) underutilizing the channel, and b) violating the rate control settings when retransmitting.

Q: For the laboratory data, is the difference in throughput between the 250-byte packet tests and the 1000-byte packet tests consistent with the difference in protocol overhead? for 250-byte packets versus 500-byte packets?

A: No, not entirely, but the discrepancy is traceable back to the rate control problem mentioned in the previous section. Using the analysis presented in the previous section, we would expect the throughput of the 250-byte packet tests to be 87% of the 1000-byte packet tests, and to be 91% of the 500-byte packet tests. The solutions for n are n=(1000+52/(1000+4*52)) and n=(500+52/(500+2*52)), respectively. Instead, we see 85% and 89%, respectively. As mentioned in the throughput analysis of the 500-byte packet tests, the laboratory data was affected to a minor degree by the interaction between the system clock granularity and the rate control settings. This effect reduces the rate at which raw data is transmitted for 250-byte packets by 2% from the 1000-byte packet case. This accounts for the 2% throughput discrepancy between the 1000-byte case and the 250-byte case. The data rate reduction for 250-byte packets compared to 500-byte packets is also 2%, accounting for the remaining discrepancy.

(For 125-byte packets, the reduction from the 1000-byte packet case is 4%, and for 500-byte packets, the reduction in throughput is completely attributable to the difference in header overhead.)

Q: Is the throughput performance at relatively high bit error rates consistent between the laboratory data and the experiment data?

A: Yes, at a BER of 10^{-5}, both the laboratory data and the experiment data show throughput that is 96% of the throughput of the lowest error cases.

Conclusions:

The rate control problems identified in the analysis of 500-byte throughput affect the 250-byte packet tests as well. Of the two problems identified, one has been resolved and one is still under investigation.

Throughput at relatively high BERs ($\sim 10^{-5}$) is high: 96% of the value of the data points at the lowest BERs, for both the laboratory data and the experiment data.

6.2.1.4 Throughput analysis of 125-byte packet tests

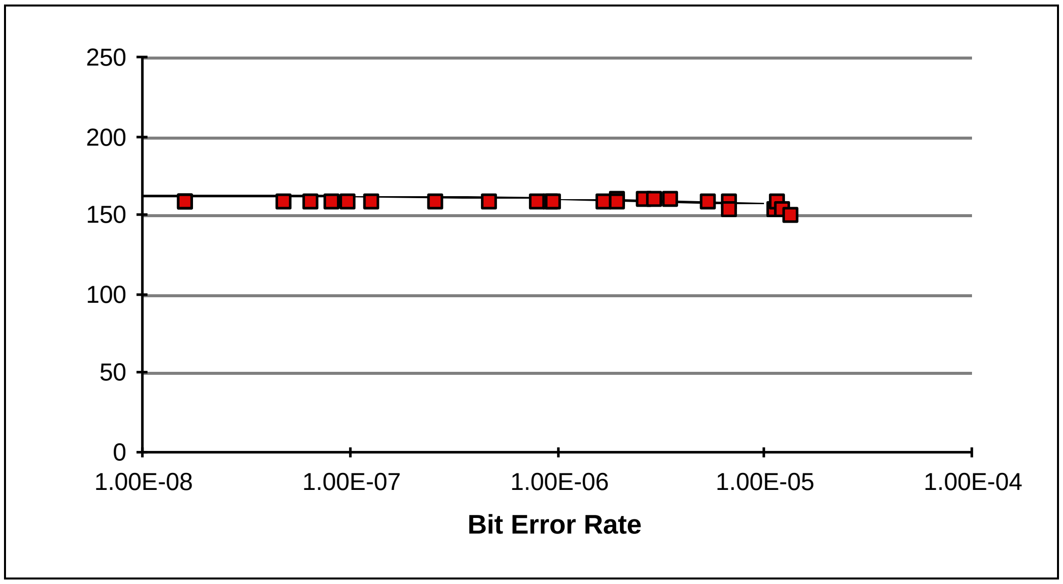
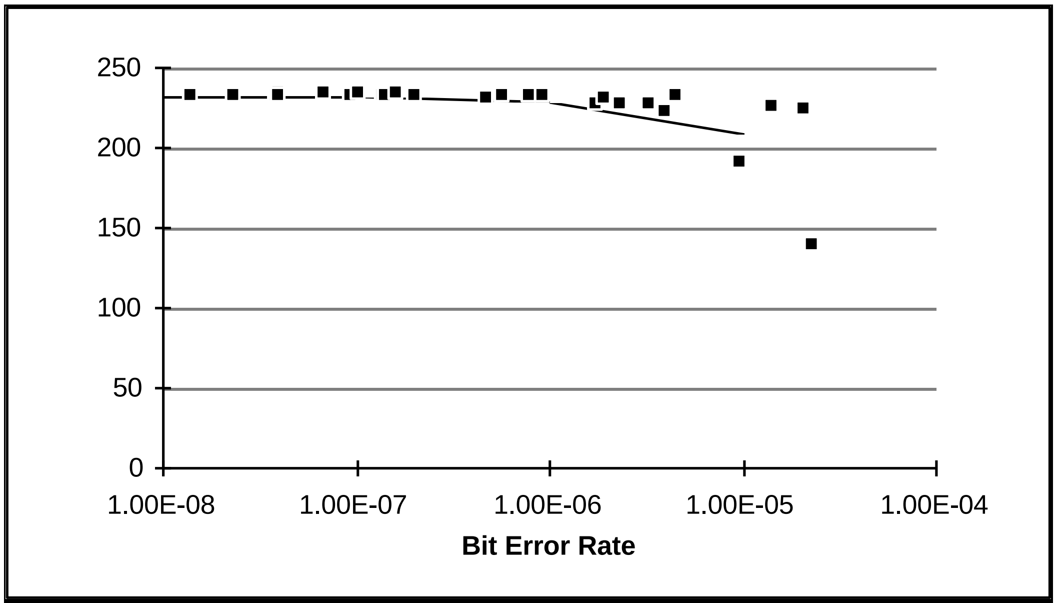


Figure 16. SCPS-TP Throughput Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data

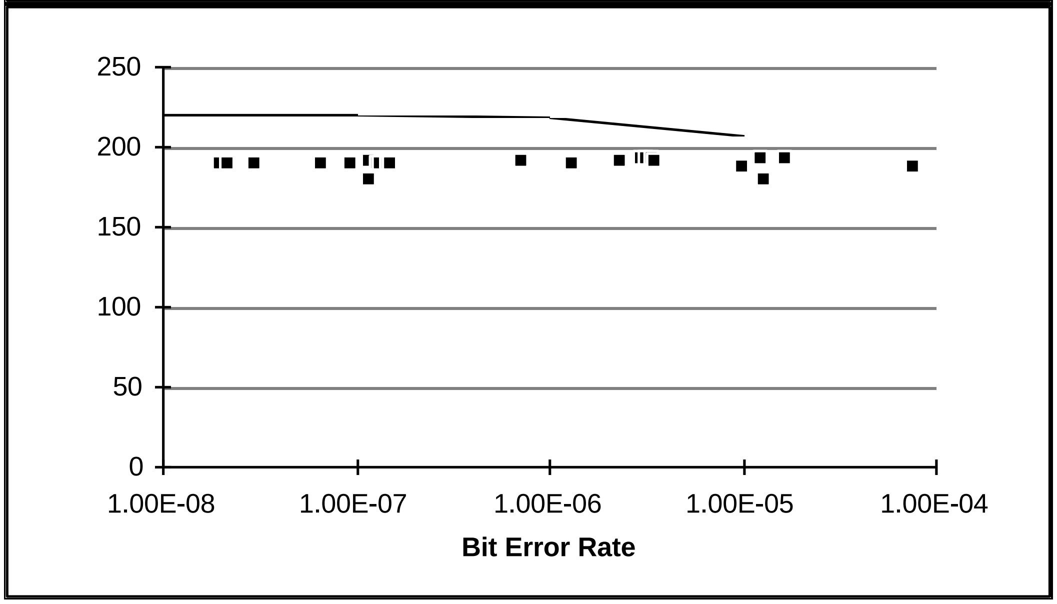
Figure 16 shows throughput of SCPS-TP versus bit-error rate for the laboratory data and for the bent-pipe test data with 125-byte packets. The throughput of the experiment data is consistent with that of the laboratory data, since the rate control parameters for the experiment data were empirically set to yield the desired rate. We also note that the throughput is relatively flat to an error rate of 10^{-5}. The experiment data exhibits throughput at a BER of $1.1x10^{-5}$ that is approximately 96.8% of the throughput experienced at error rates better than 10^{-7}. The throughput of the laboratory data at 10^{-5} BER is 97.5% of the throughput experienced at error rates better than 10^{-7}.

Conclusions:

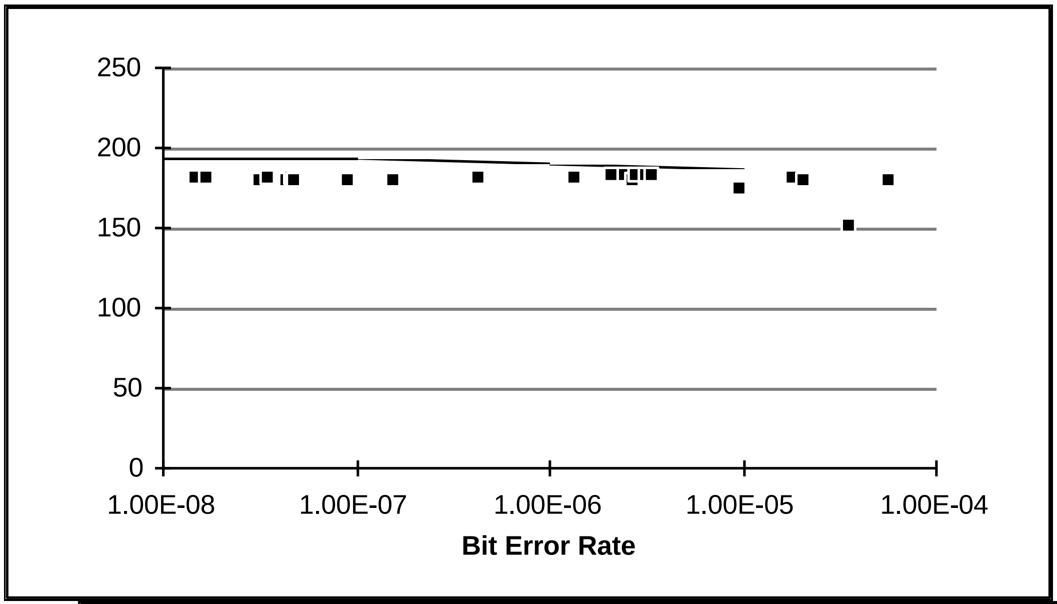
The laboratory data and experiment data yield consistent throughput performance for 125-byte packets. For both the laboratory and experiment data, throughput at a BER of 10^{-5} is greater than 96.5% of the throughput at 10^{-7}.

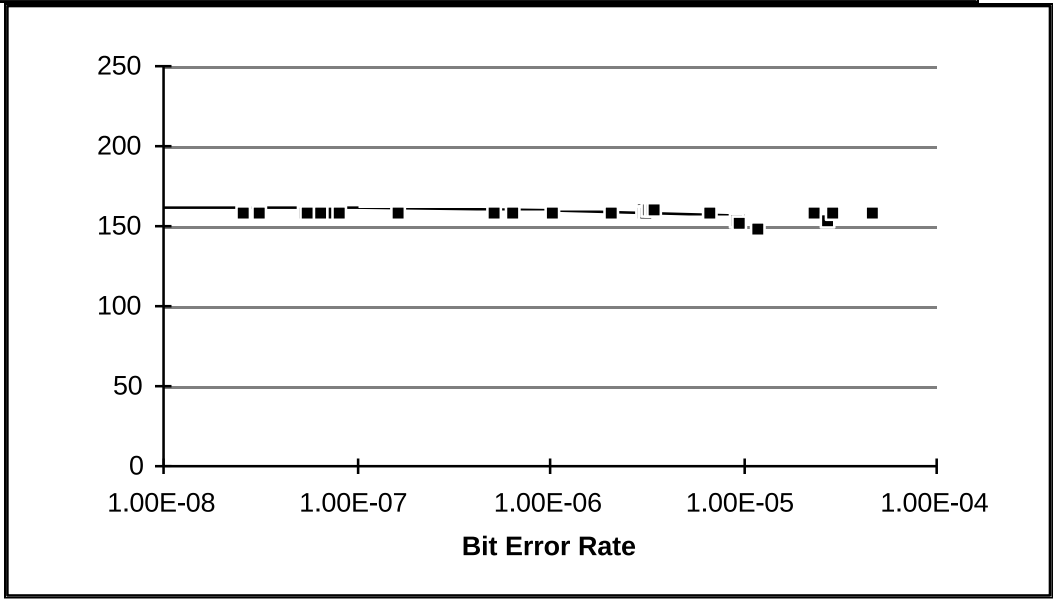


250
200
150
100
50
0
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate



250
200
150
100
50
0
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate





6.2.2 Data Channel Utilization

Data channel utilization is an indication of the ability of the protocol to maintain a constant load on the network. If the load is lower than requested, network capacity is being wasted. If the load is higher than requested, other network users could be experiencing congestion.

This section presents the bent-pipe experiment data channel utilization results for each packet size. Each packet size is presented on a separate graph, with the laboratory channel utilization information plotted for reference. We summarize the results, present a discussion of the results when appropriate, and then draw conclusions. Readers not interested in a detailed discussion of each graph may wish to read the introductory paragraph of each subsection, then skip to the conclusions at the end of each subsection.

6.2.2.1 Data channel utilization analysis of 1000-byte packet tests

Figure 17 shows the data channel utilization of SCPS-TP versus bit-error rate for the laboratory data (shown as a thin, solid line) and the bent-pipe test data. The laboratory data is shown as the mean of five runs at each of the following four bit-error rates: 10^{-8}, 10^{-7}, 10^{-6}, and 10^{-5}. The bent-pipe test data is shown as individual samples. The data channel utilization for each sample is measured by dividing the total data transmitted during the course of the test by the duration of the test, then scaling by the channel capacity. The bit-error rate (BER) for each sample is estimated as described in Section 6.3. For this graph, each SCPS-TP segment (packet) contained 1000 bytes of user data. We see that the data channel utilization of SCPS-TP is relatively constant BERs of 10^{-8} and $5x10^{-6}$. At BERs greater (i.e., worse) than $5x10^{-6}$, the data channel utilization "rolls off" steeply.

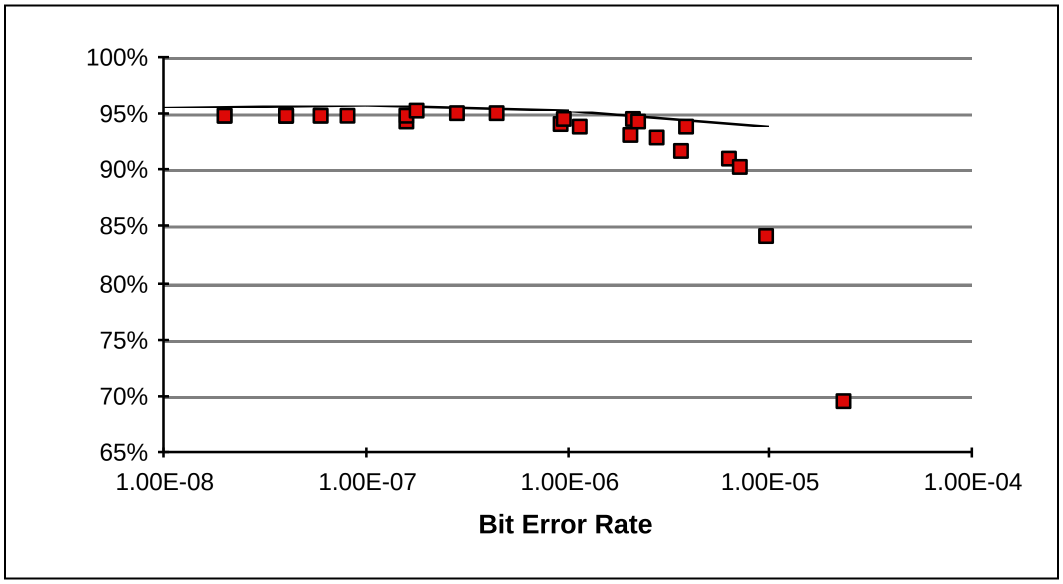


Figure 17. SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Questions:

Q: As stated in the introduction to this section, one of the reasons for measuring data channel utilization is to determine whether the protocol maintains a constant load on the network.

However, we see in Figure 17 that the channel utilization decreases with increasing BER. What does this indicate and why does it occur?

A: The roll-off in data channel utilization indicates that the sender is not "keeping the pipe full." That is, the sender is not maintaining a steady stream of transmissions and retransmissions.

The data show two causes for decreasing channel utilization that are supported by this experiment's data both relate to memory availability. The first cause is a previously-mentioned bug in the memory buffer code. Consider the two data points on Figure 17 that exhibit the lowest data channel utilizations: 86% and 71%, respectively. When we examined the log files for the 86% test, we noted that the receiver logged memory buffer errors during this run. These memory buffer errors caused the receiver to indicate that it had less memory than it actually did, which in turn exerted flow control on the sender. This caused the data channel utilization (and the user data throughput) to drop significantly. Although we do not have the log files for the 71% test, we suspect that this test experienced the same memory buffer problems as the previous one. Subsequent to the experiment testing, this memory problem was identified and repaired.

The second cause for decreasing data channel utilization is illustrated by the points on Figure 17 that correspond to utilizations of approximately 93% and BERs of approximately $6x10^{-6}$ and $7x10^{-6}$, respectively. This decrease is due to having insufficient buffer memory at the sender and receiver. The buffer space at the receiver is approximately 69 K bytes, and the buffer space at the sender is slightly more than that. At a data rate of 255 Kbps and a round-trip delay of 0.5 seconds, the "bandwidth-delay product" of the network is 255 Kbps / 8 bits/byte * .5 seconds, or 15.9 Kbytes. Therefore, our buffer memory is sufficient to store data for approximately 4.3 round trip times, transmitting at 255 Kbps. However, this does not appear to be sufficient, although a theoretical analysis suggests that it should be. (At a bit-error rate of 10^{-5}, of the initial 6000 packets, on the average 477 will be affected by a bit-error, assuming independent, identically-distributed errors. These 477 packets will require retransmission, and some of these may be corrupted. On the average, 38 of the 477 will require a second retransmission, and 3 of the 38 will require a third retransmission. Of the 3 retransmitted packets, one of these will be corrupted an average of every fourth run. Since we have four round trips worth of buffer space, and the probability of requiring more than four transmissions of a packet to successfully transmit it is approximately .25, we would not expect to experience persist behavior. However, it appears that this simple model is somewhat *too* simple. It does not appropriately represent the dynamics of multiple packet losses in close proximity to each other and the subsequent recovery. We performed a few laboratory experiments with buffer sizes equal to approximately 5 times the bandwidth-delay product, and saw no occurrences of persist mode. The ability to appropriately allocate buffer space is important, and we will continue to analyze this issue.)

Q: Was the load on the network that which was requested?

A: No, we experienced problems setting the rate control problems, as discussed previously. Due to the eccentricities of the high-speed serial card interfaces, we had to “tune” the parameters to achieve the desired data channel utilization. This tuning is acceptable for prototype testing, but not for eventual deployment. We have identified the problem with the rate control settings and have repaired it. However, we have not yet established a means to ensure that the problem does not recur. This is an area for further work.

Conclusions:

The data channel utilization was adversely affected by three problems: First, the previously-discussed rate control problem required us to “tune” the rate settings to achieve the desired channel utilization. Second, the previously-discussed memory buffer problem caused some of the runs to underutilize the network. Finally, the amount of buffer memory in the sender and receiver was insufficient to permit continuous transmission at high BERs. The first two problems have been identified and corrected. The third problem merely requires a buffer configuration change that is specific to the particular network in test, but requires some further analysis to adequately model the buffer requirements as a function of bit-error rate.

6.2.2.2 Data channel utilization analysis of 500-byte packet tests

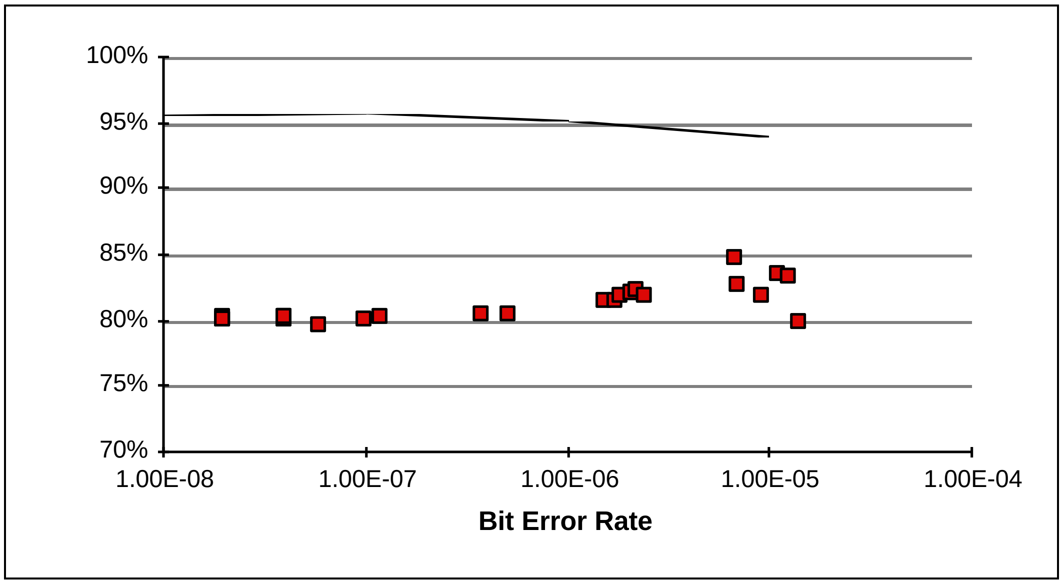


Figure 18. SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figure 18 shows data channel utilization of SCPS-TP versus bit-error rate for the laboratory data and for the bent-pipe test data. The difference between this graph and Figure 17 is that each SCPS-TP segment shown in Figure 18 carried 500 bytes of user data, rather than 1000 bytes of user data. We see that the data channel utilization for the experiment data is lower than that of the laboratory data, caused by the previously-discussed rate-control settings. We also note that the data channel utilization increases between 10^{-6} and 10^{-5}.

Q: Why does the data channel utilization (and therefore data transmission rate) increase between 10^{-6} and 10^{-5}?

A: Although this behavior is still under investigation, we believe that this is the result of the violation of the rate control settings by retransmission logic. This problem was first discussed in Section 6.2.1.2. The increase in data transmission rate with increasing errors is an indication that the retransmission logic was violating the rate control parameters. An examination of the software leads us to believe that the effect could also have increased the rate at which new transmissions were shipped (the rate control parameters were temporarily increased).

Conclusions:

The laboratory measurements of data channel utilization indicate that the protocol is performing as expected. The performance indicated by the experiment data show no previously-unidentified problems, and provide some insight into the potential cause of the rise in throughput for increasing bit-error rates.

6.2.2.3 Data channel utilization analysis of 250-byte packet tests

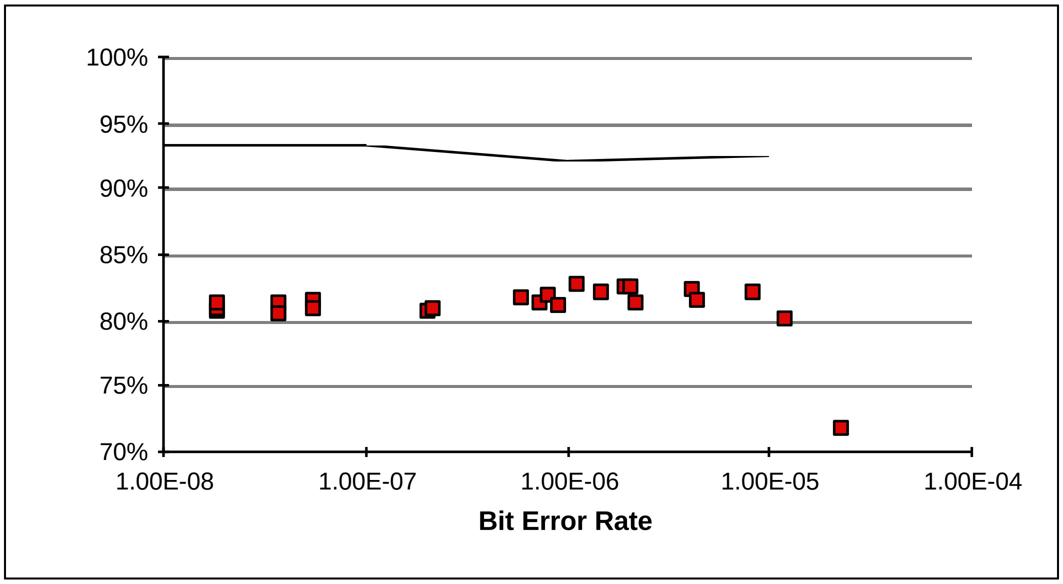


Figure 19. SCPS-TP Data Channel Utilization Versus Bit-Error rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figure 19 shows data channel utilization of SCPS-TP versus bit-error rate for the laboratory data and for the bent-pipe test data with 250-byte packets. The data channel utilization for the experiment data is approximately 12% lower than that of the laboratory data. As with other examples of the retransmission logic/rate control problem, the experiment data shows an increase in data channel utilization between approximately 10^{-6} and 10^{-5}.

Q: Why does the laboratory data appear to dip at 10^{-6} and then increase at 10^{-5}?

A: There was one data run of the 10^{-6} tests that had a lower data channel utilization than the other runs. As a result, the mean value for the 10^{-6} data point is 0.5% lower than that of the 10^{-5} data point. However, we calculated the 90% confidence intervals for both the 10^{-6} and the 10^{-5} data points, and they overlap, indicating that the difference is not statistically significant.

Q: Why does the experiment data point at ~2x10^{-5} exhibit such a low data channel utilization?

A: An examination of the logs for this run indicates that it experienced the memory buffer problem described in Section 6.2.1.1. This problem has been corrected.

Conclusions:

The performance indicated by the experiment data show no previously-unidentified problems.

6.2.2.4 Data channel utilization analysis of 125-byte packet tests

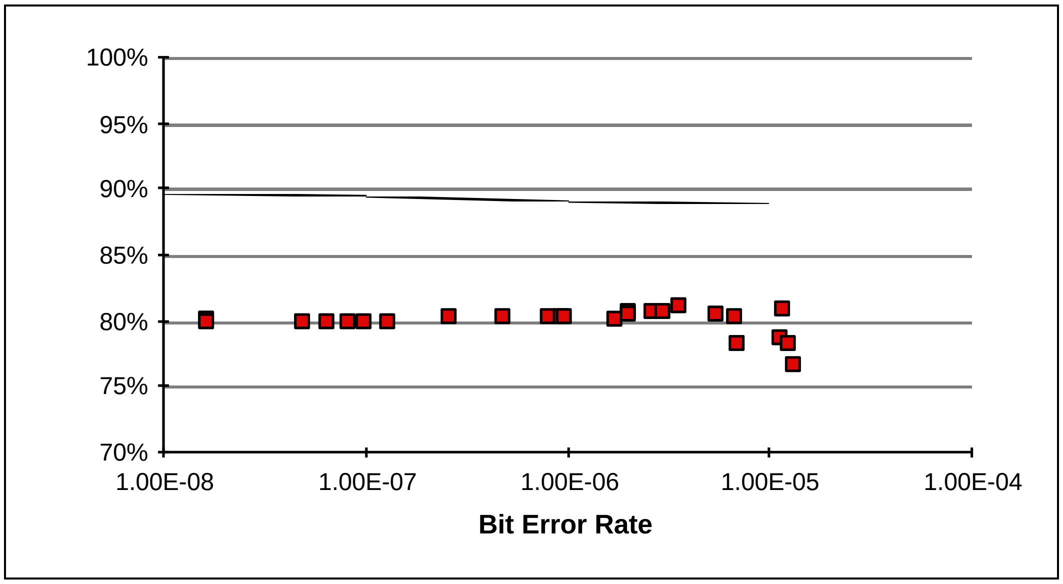


Figure 20. SCPS-TP Data Channel Utilization Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and BenPipe Test Data

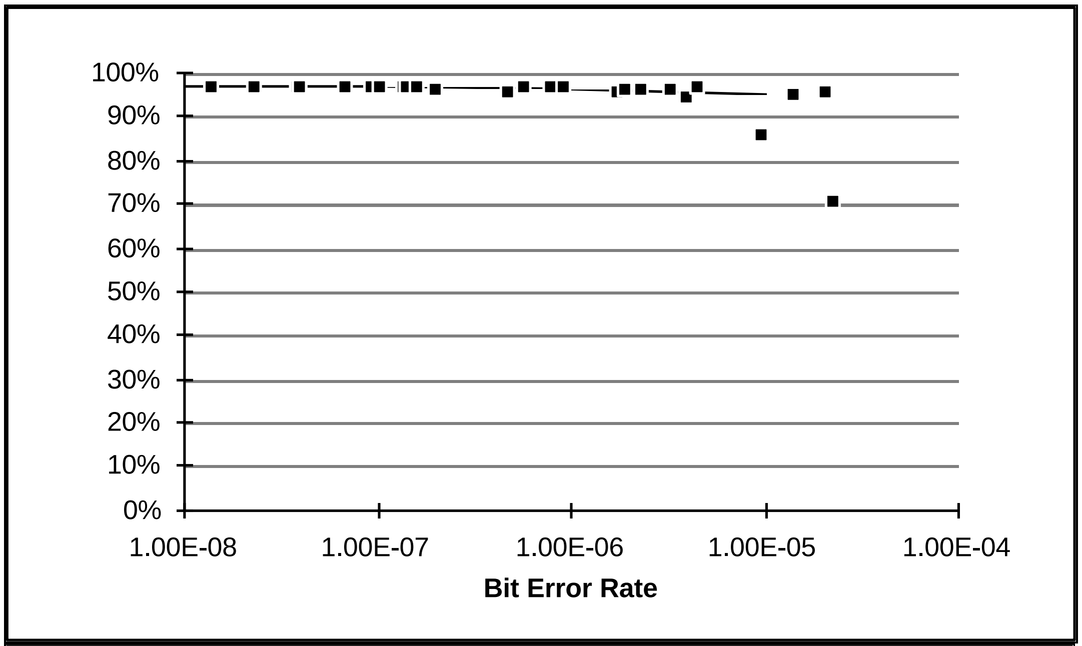
Figure 20 shows data channel utilization of SCPS-TP versus bit-error rate for the laboratory data and for the bent-pipe test data with 125-byte packets. The data channel utilization for the experiment data is approximately 10% lower than that of the laboratory data. As with other examples of the retransmission logic/rate control problem, the experiment data shows an increase in data channel utilization between 10^{-6} and 10^{-5}.

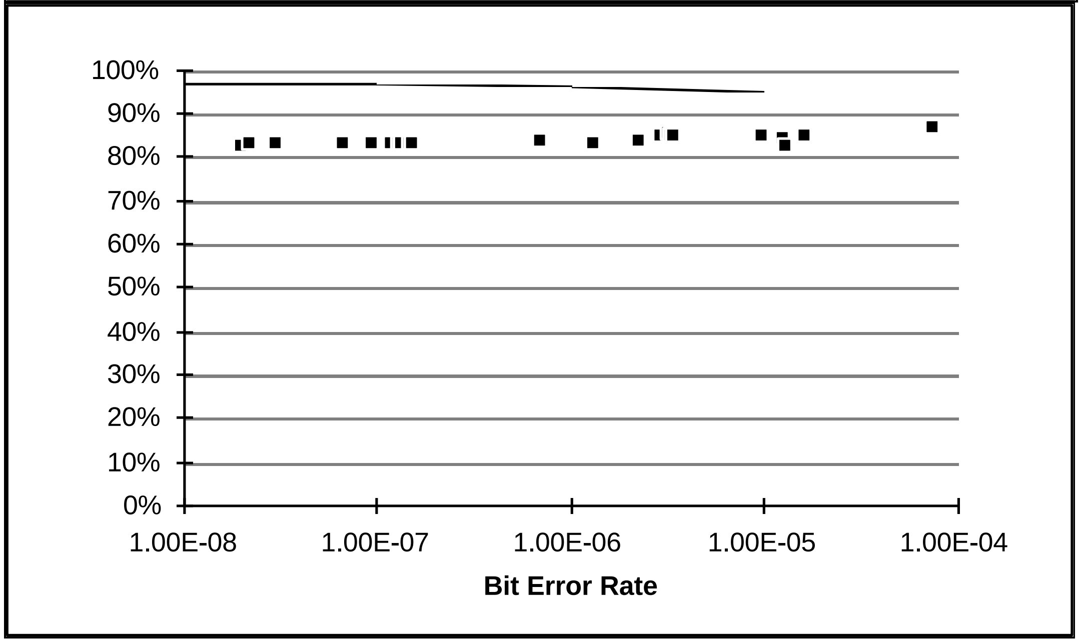
Q: In Section 6.2.1.4, the throughput of the experiment data and laboratory data is almost identical for 125-byte packets. What accounts for the large discrepancy in the data channel utilizations at this packet size?

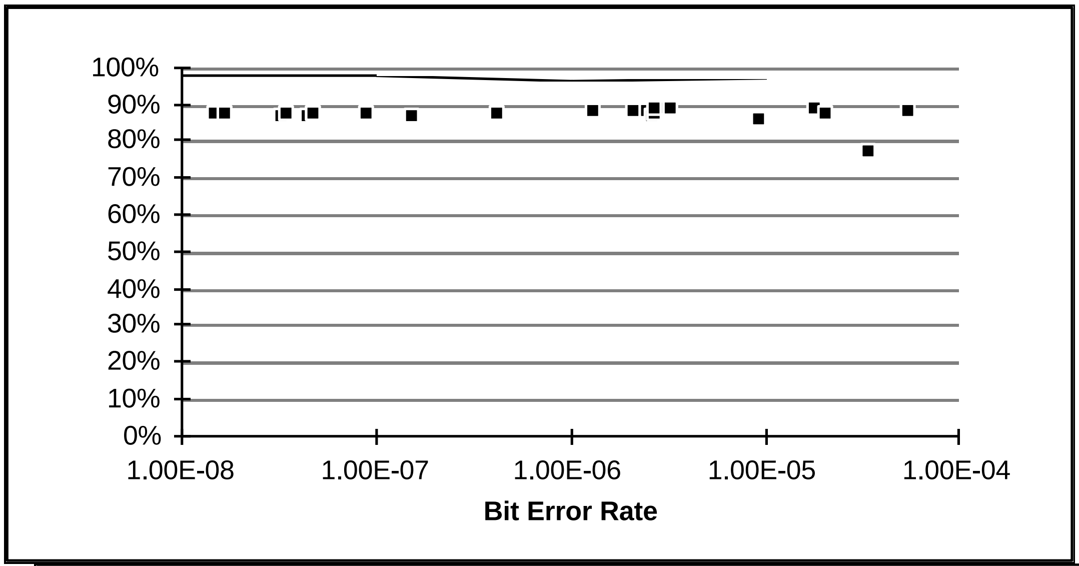
A: As packet size decreases, the beneficial effects of SCPS-TP header compression become more apparent. The bent-pipe experiment was conducted with header compression on, while the laboratory data was collected with header compression off. The lower data channel utilization (for equivalent throughput) is attributable to header compression. (Note: had the rate control settings been operating as expected, the effect would have been equivalent data channel utilizations, and higher user data throughput when header compression is in use. However, for the reasons discussed before, the rate control parameters did not match.)

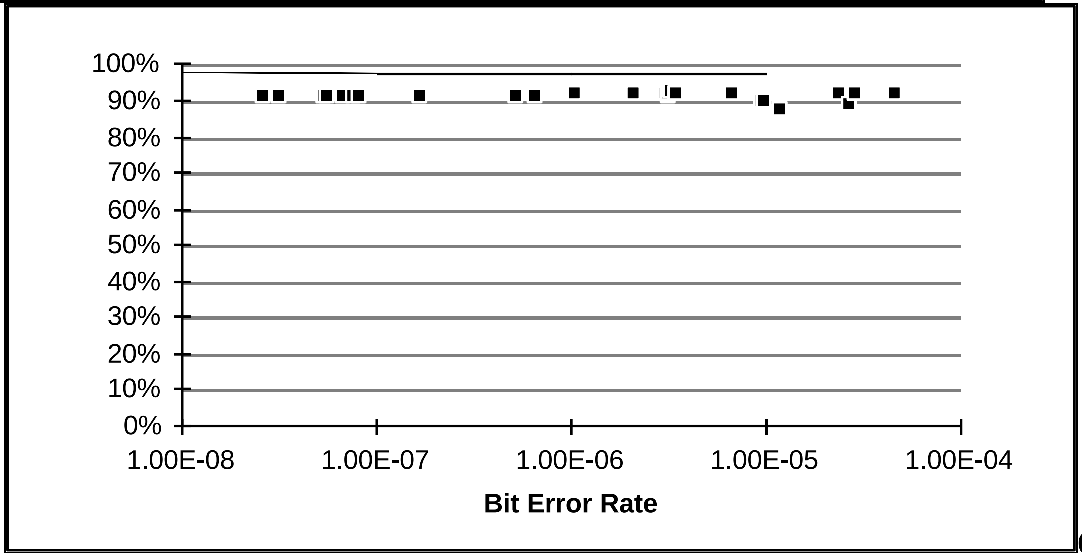
Conclusions:

The performance indicated by the experiment data show no previously-unidentified problems.


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate





6.2.3 Acknowledgment Traffic Results

An examination of acknowledgment traffic is important if the protocol is to be operated over unbalanced-rate links, in which there may not be much bandwidth available for acknowledgment transmission.

This section presents the bent-pipe experiment acknowledgment traffic results for each packet size. Each packet size is presented on a separate graph, with the laboratory acknowledgment traffic information plotted for reference. Due to the similarity of the graphs and discussion, we have not created separate subsections for each packet size.

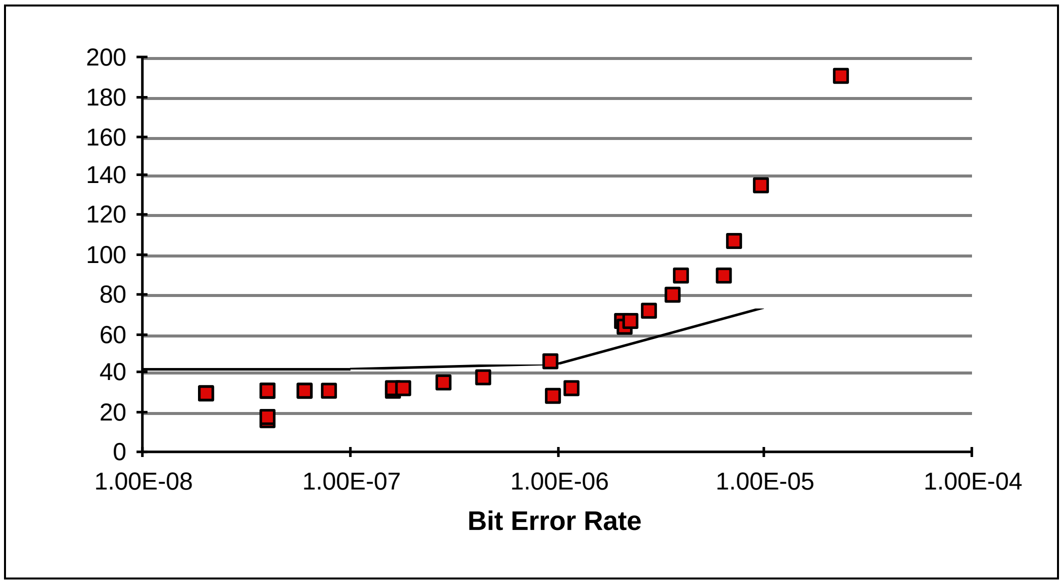


Figure 21. SCPS-TP Acknowledgment Traffic Versus Bit-Error rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data

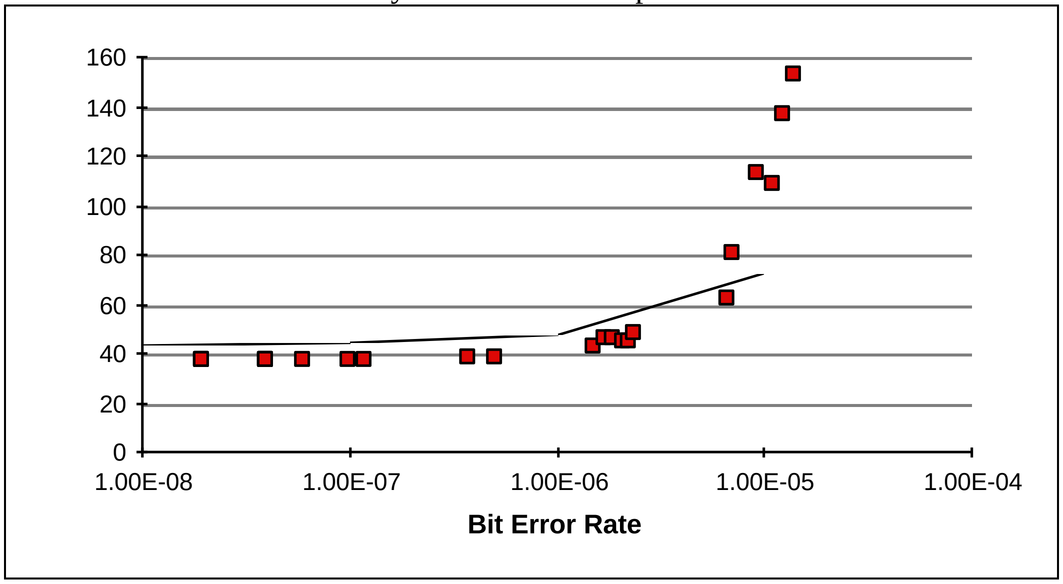


Figure 22. SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data

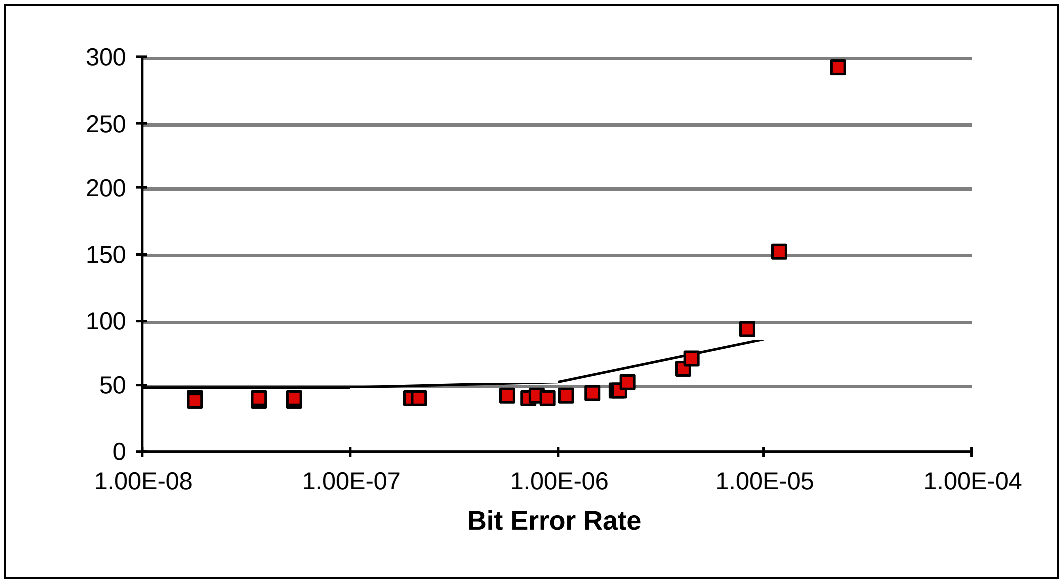


Figure 23. SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data

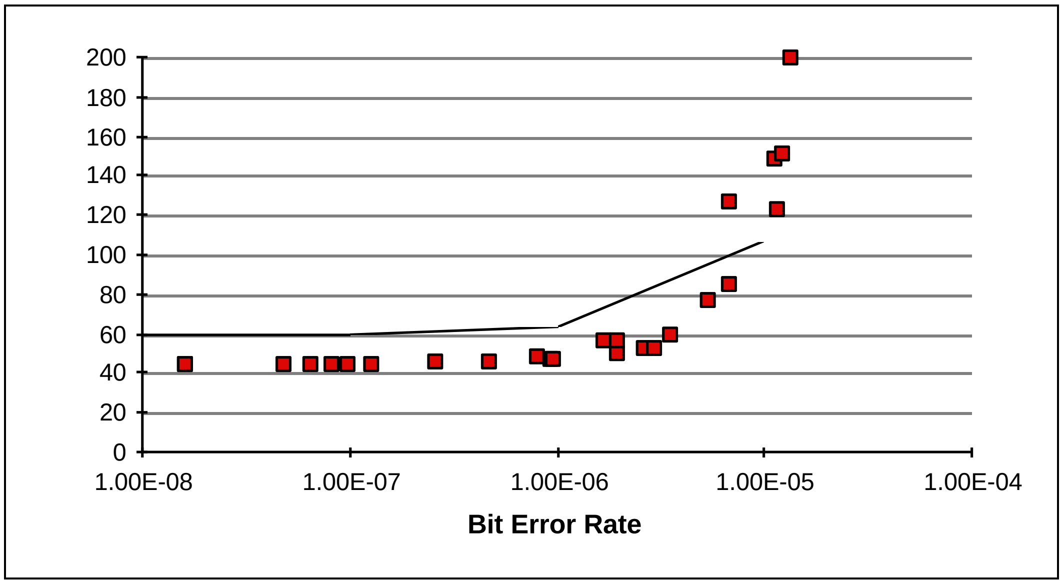


Figure 24. SCPS-TP Acknowledgment Traffic Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figures 21 through 24 show the acknowledgment traffic generated by the SCPS-TP receiver versus bit-error rate for the laboratory data (shown as a thin, solid line) and for the bent-pipe test data. The laboratory data is shown as the mean of five runs at each of the following four bit-error rates: 10^{-8}, 10^{-7}, 10^{-6}, and 10^{-5}. The bent-pipe test data is shown as individual samples, with acknowledgment traffic as measured, and bit error rate estimated as described in Section 6.3.

The graphs show that as the bit-error rate increases from 10^{-8} to 10^{-6}, both the laboratory data and the experiment data show minor increases in the amount of acknowledgment data generated. This increase is due to the transmission of Selective Negative Acknowledgments (SNACKs) options, that are added to normal outbound acknowledgments to request immediate retransmission of one or more missing packets. Note also that as the bit-error rate increases from 10^{-6} to 10^{-5}, both the laboratory data and experiment data show significant increases in the amount of acknowledgment traffic generated, with the volume of acknowledgment traffic for the experiment data set increasing by more than a factor of three. The laboratory data increases by less than a factor of two.

Q: Why is there less acknowledgment traffic for the experiment data than for the lab data at low error rates?

A: The experiment was conducted with header compression enabled, while the laboratory data was taken with header compression disabled.

Q: Why so much more acknowledgment data evident in the experiment data than in the lab data at high error rates?

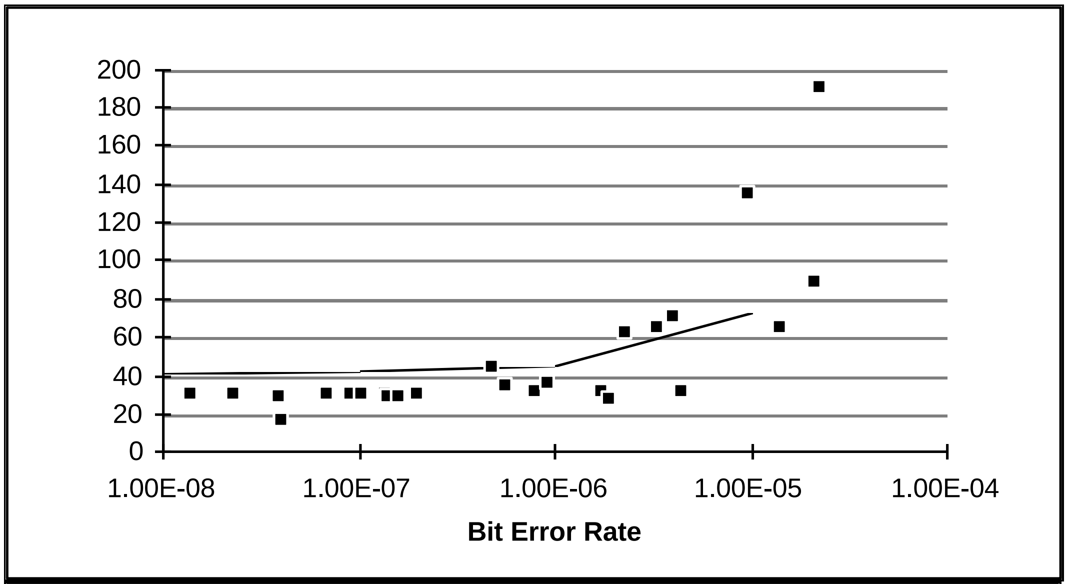
A: Lab uses new acknowledgment strategy designed to reduce acknowledgment traffic. The protocol configuration used in generating the experiment data can send acknowledgments as a result of a number of different kinds of events: increases in window size, closing holes in the out of sequence queue, etc. The protocol configuration that generated the laboratory data sends fewer acknowledgments. This approach has not yet been tested at error rates greater than 10^{-5}, but it is promising.

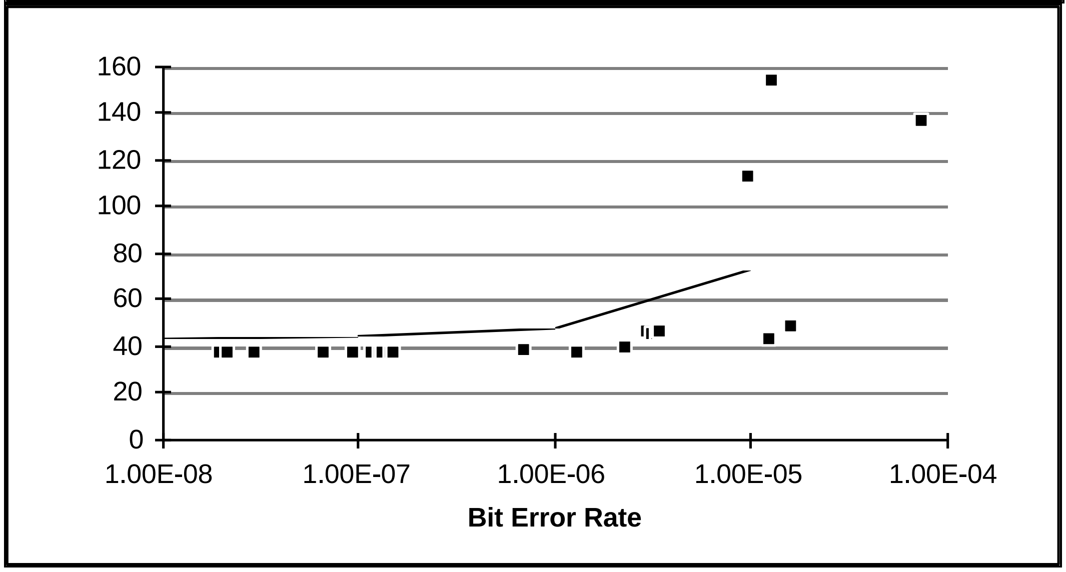
Q: Why does the amount of acknowledgment data increase so much as the error rate approaches 10^{-5}?

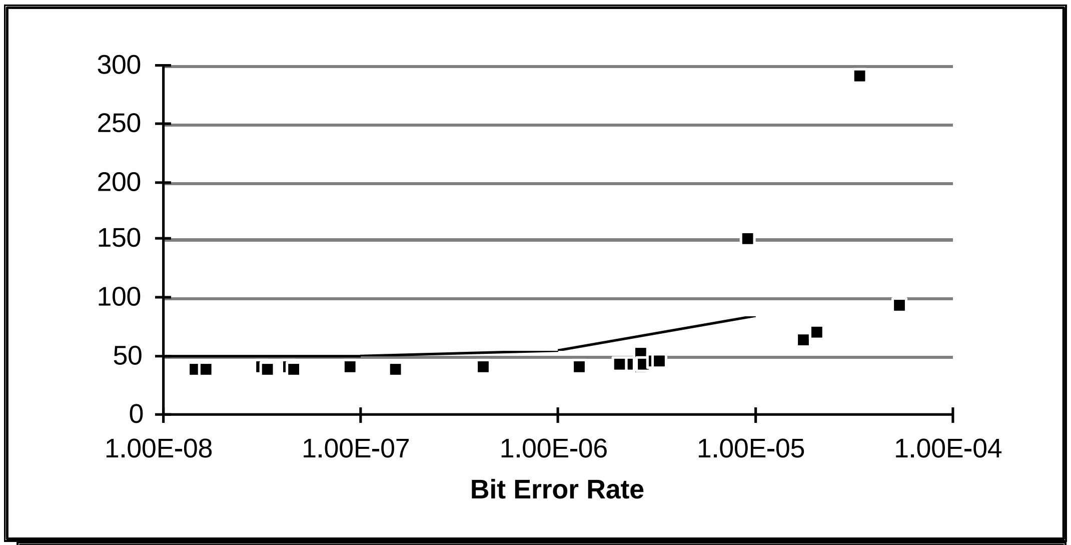
A: There are a number of triggers that can cause acknowledgments to be sent - receipt of out of sequence data that opens a new “hole” in the receiver’s out-of-sequence queue, closing a similar hole, an increase in the amount of receive buffer memory available, etc. Many of these triggers fire when the error rate is high. We are currently working to identify the proper set of triggers to maximize user data throughput while holding the acknowledgment traffic to a minimum.

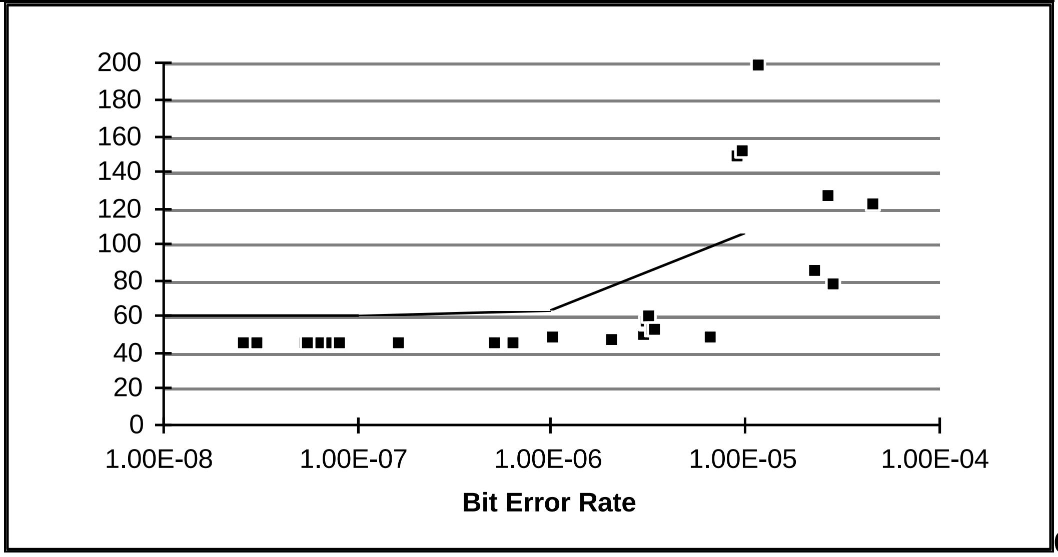
Conclusions:

The modifications to the acknowledgment strategy that are shown in the laboratory data reduce the amount of acknowledgment traffic generated and do not appear to adversely affect user data throughput. These should be evaluated at higher error rates.









6.2.4 Bit Efficiency Results

An examination of bit efficiency is important if the protocol is to be operated over restricted-capacity links, which may be sensitive to the amount of protocol header overhead.

This section presents the bent-pipe experiment bit efficiency results for each packet size. Each packet size is presented on a separate graph, with the laboratory bit efficiency information plotted for reference. Due to the similarity of the graphs and discussion, we have not created separate subsections for each packet size.

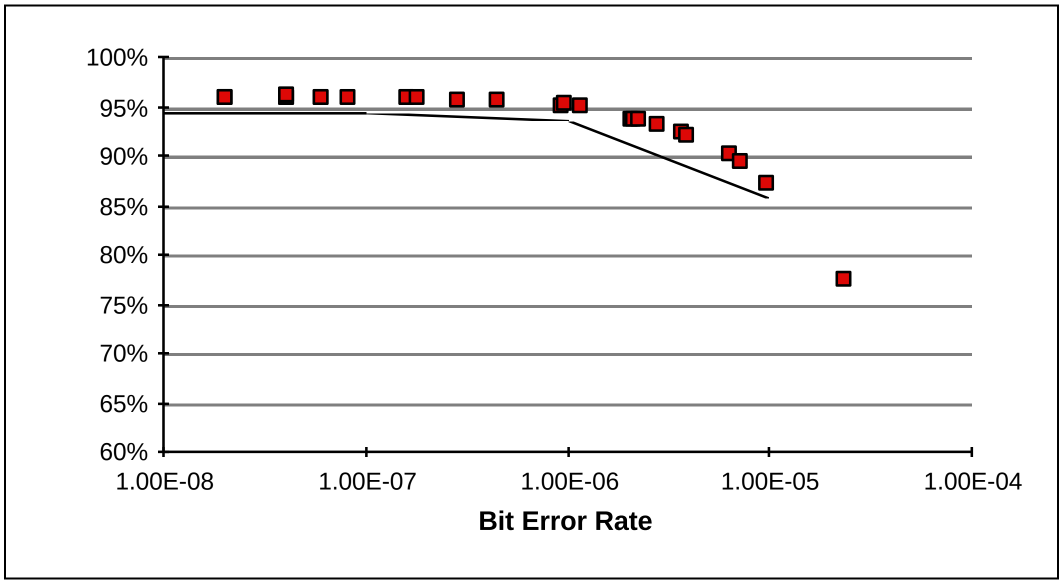


Figure 25. SCPS-TP Bit Efficiency Versus Bit-Error Rate for 1000-Byte Packets: Laboratory Data and Bent-Pipe Test Data

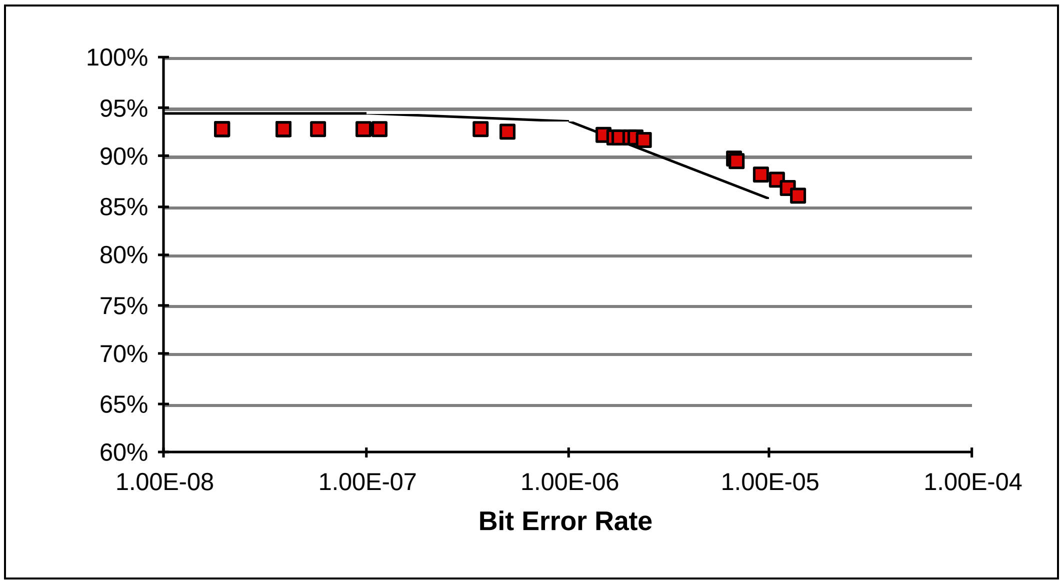


Figure 26. SCPS-TP Bit Efficiency Versus Bit-Error Rate for 500-Byte Packets: Laboratory Data and Bent-Pipe Test Data

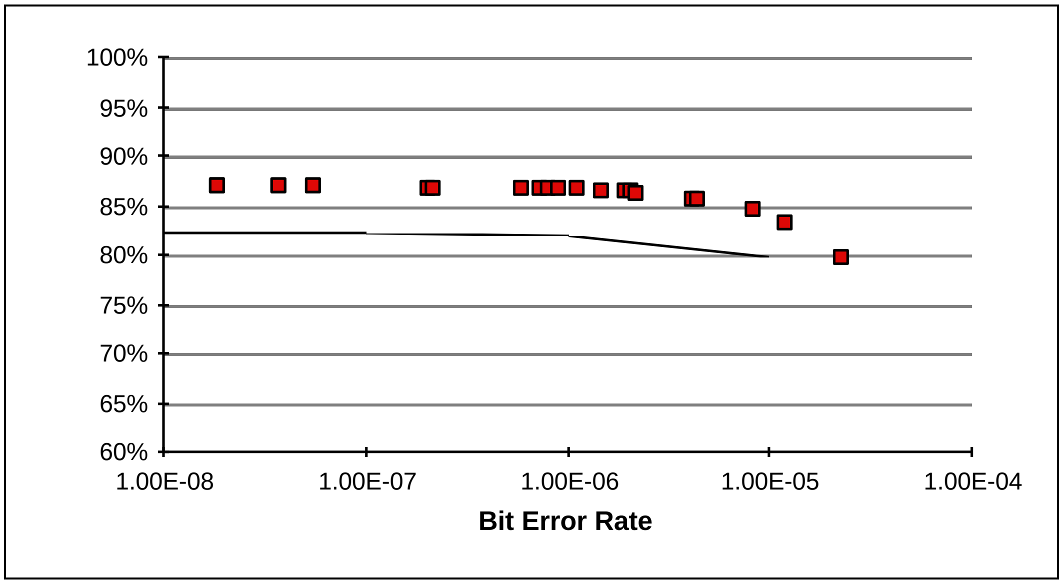


Figure 27. SCPS-TP Bit Efficiency Versus Bit-Error Rate for 250-Byte Packets: Laboratory Data and Bent-Pipe Test Data

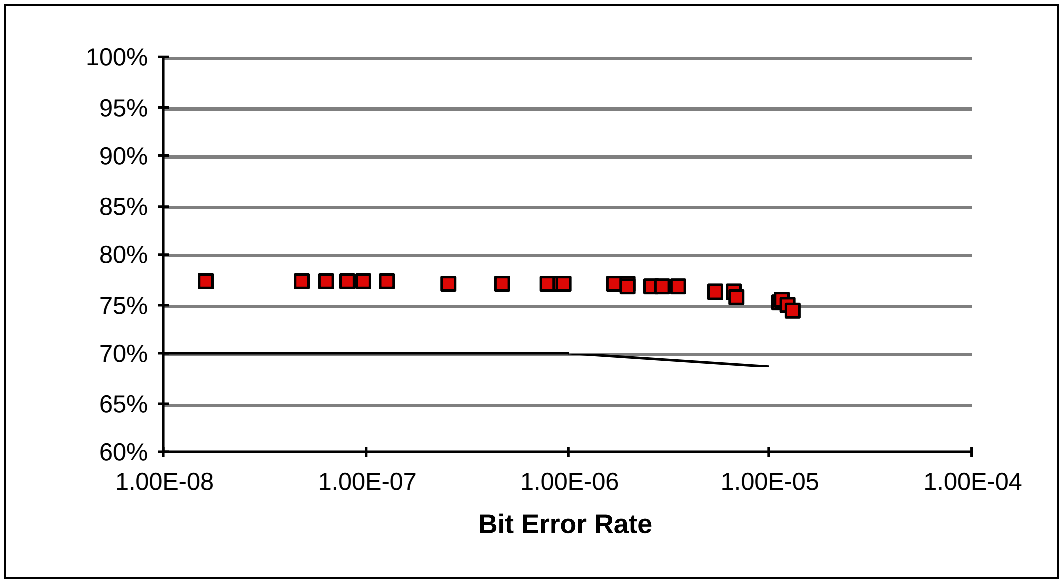


Figure 28. SCPS-TP Bit Efficiency Versus Bit-Error Rate for 125-Byte Packets: Laboratory Data and Bent-Pipe Test Data

Figures 25 through 28 show the bit efficiency of SCPS-TP versus bit-error rate for the laboratory data (shown as a thin, solid line) and for the bent-pipe test data. The laboratory data is shown as the mean of five runs at each of the following four bit-error rates: 10^{-8}, 10^{-7}, 10^{-6}, and 10^{-5}. The bent-pipe test data is shown as individual samples, with bit efficiency calculated as the total user data (six million bytes) divided by the total number of bytes sent by both the initiator and the responder. The bit error rate is estimated as described in Section 6.3.

The graphs show that as the bit-error rate increases, bit efficiency decreases. This is due to retransmissions, which are considered overhead, rather than user data. Note that in all cases, the experiment data has a higher bit efficiency than the laboratory data. This is because SCPS-TP header compression was in use for the experiment data, but not for the laboratory data. The difference in bit efficiency becomes much more pronounced as packet sizes are reduced. Note that all of this data was taken with IP as the network protocol. IP headers are 20 bytes in length. SCPS-NP headers can be as short as five bytes, which would further improve bit efficiency.

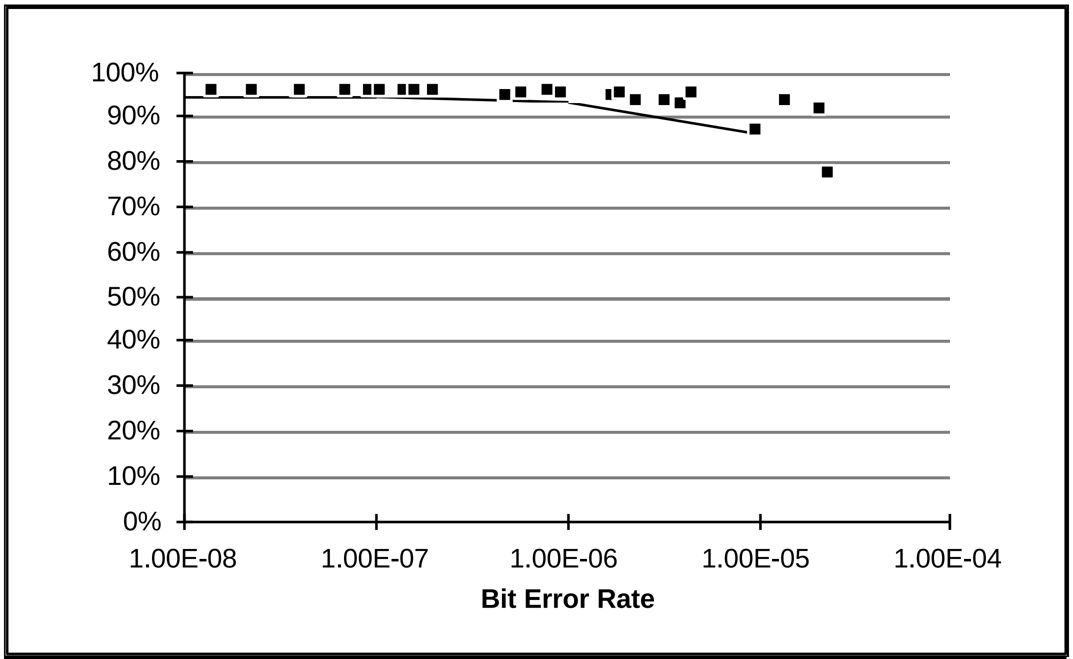
There was a goal, but not a firm requirement, that SCPS bit efficiency be 90% or better when no bit-errors are present. An examination of the graphs shows that this goal is fulfilled for packet sizes of 1000 bytes and 500 bytes, both when SCPS-TP header compression is in use and when it is not in use. Note that when SCPS-TP header compression is in use, the experiment data indicates that at a BER ² 7.1x10^{-6}, the 1000-byte packet configuration operates at bit efficiencies ³ 90%. For 500-byte packets, the experiment data indicates that at a BER ² 6.6x10^{-6}, the 1000-byte packet configuration operates at bit efficiencies ³ 90%.

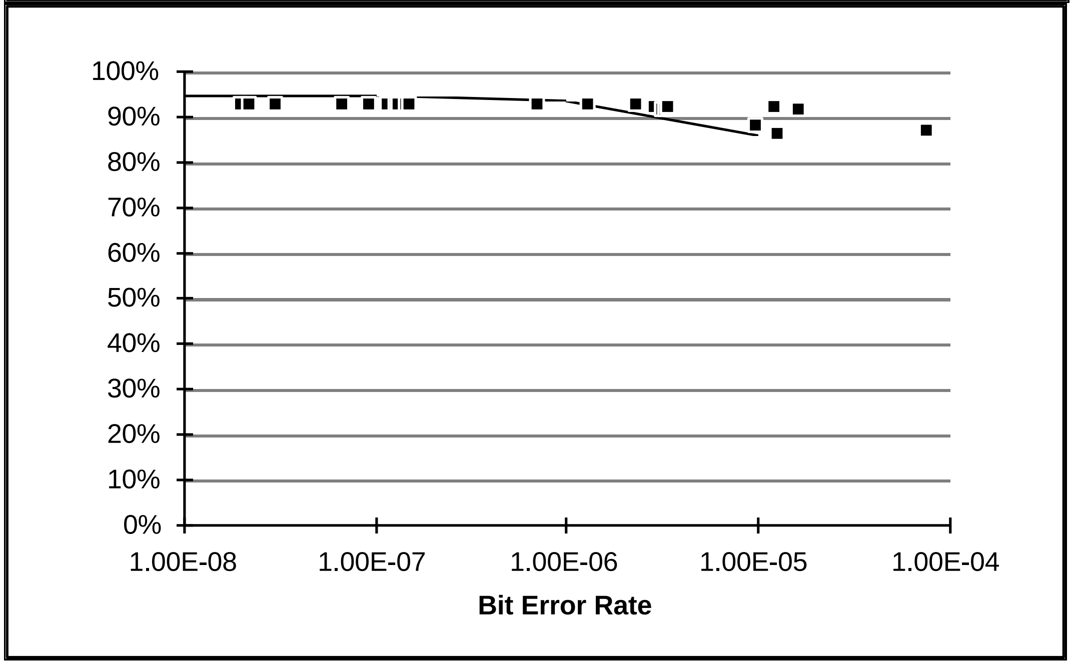
Note that if the SCPS-NP protocol were used at the network layer instead of IP, and were configured to use a 5-byte protocol header, the 90% threshold would be extended for 500-byte packets to 1.23x10^{-5}, and for 250 byte packets to 8.3x10^{-6}. The threshold remains at 7.1x10^{-6} for 1000-byte packets, and 125 byte packets do not meet the 90% bit efficiency mark at zero errors.

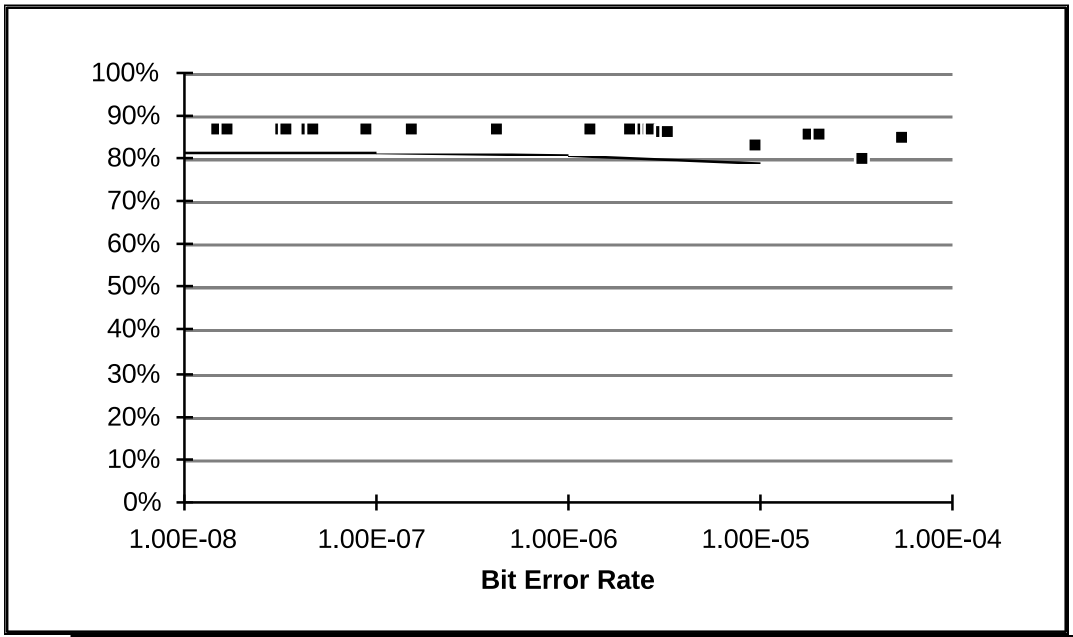
Conclusions:

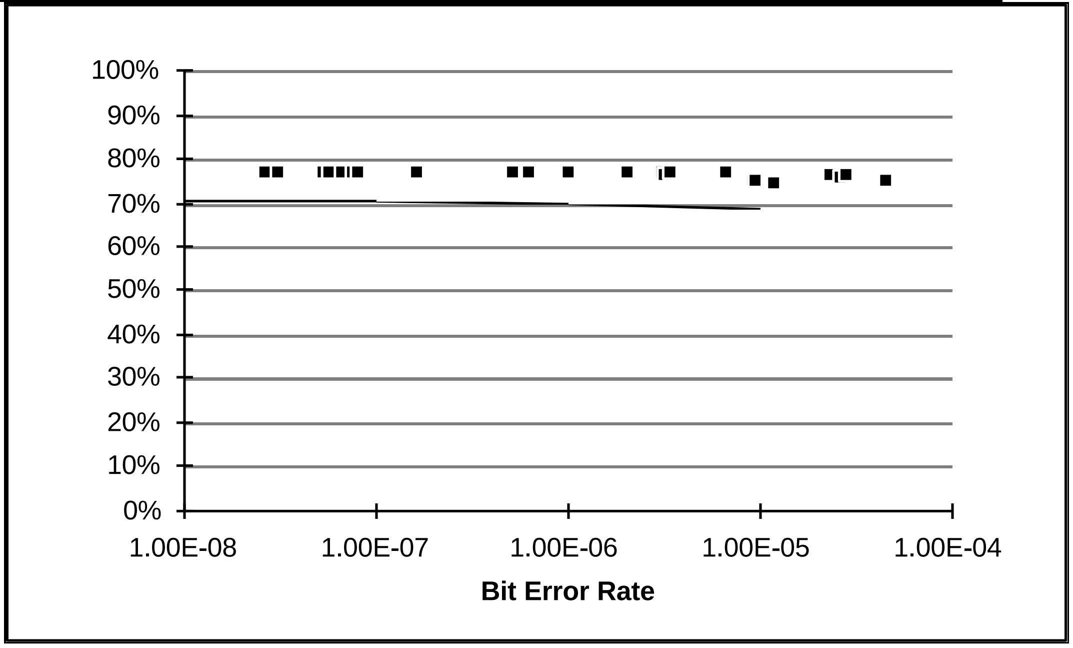
SCPS-TP header compression provides a measurable improvement in protocol bit-efficiency, up to seven percent for 125-byte packets. The SCPS-NP can further improve bit-efficiency, although it was not used in this test.

The goal of 90% or better bit efficiency at zero errors can be met by SCPS-TP over IP for 1000-byte packets and 500-byte packets.


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate


100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
1.00E-08
1.00E-07
1.00E-06
1.00E-05
1.00E-04
Bit Error Rate

6.3 LINK QUALITY ANALYSIS

The experiment design required that the link quality be known during the data runs. Link quality was measured in terms of the bit error rate on the user data link, i.e., the 256 Kbps link. Since it was not possible to simultaneously measure the BER during the data runs, BERTs were made at the beginning of each link session and then periodically throughout the contact. If a series of runs required artificially induced errors, the necessary adjustment was made and then a BERT was run before the data runs. All link quality measurements were recorded in the run logs and are tabulated in Appendix A, Table A-4. For each BERT, the following data was recorded: Time in GMT, number of errors, antenna azimuth and elevation, slant range in nautical miles, signal strength in dBm, and appropriate comments. The measurement procedure is described in Appendix C.

The BERT yielded the number of errors in five minutes for the total user data link (SGLS channel 9, 1.7 Mhz subcarrier uplink and downlink). This value was then converted to BER using equation B.3-1 shown in Appendix B. Note that the BERT was run at 256 Kbps, the channel maximum data rate, even though the data rate used for the experiment runs was 255 Kbps. The experiment data rate was constrained by the high speed serial interface cards used in the SCPS workstations, while the BER test set data rate was constrained by the set's available data rates. The 1 Kbps difference was not considered significant for this experiment.

The original experiment design required a specific link bit error rate for each run. After the experiment configuration was established, it was found that the link quality could not be controlled with the precision first thought possible. However, since the protocol was designed to operate consistently over a wide continuum of error rates, it was decided that it was not necessary to make the data runs at precisely predetermined points, as long as the error rate could be reliably ascertained at the time each data run was made. This approach was considered acceptable because the M-22 link was reported to be stable and change predictably with range. This supposition was generally borne out during the experiment; the link quality usually remained high until the antenna beam encountered obstructions such as trees or buildings.

The data runs were made with both natural errors and artificial errors. Natural errors were simply those random errors that occurred naturally when the ground station was configured for normal operations. Artificial errors were introduced by decreasing the uplink modulation index. This method was chosen rather than the use of attenuators in order to avoid data losses due to threshold conditions in the receiver and bit synchronizer circuitry; i.e., it was deemed desirable to retain a consistent signal+noise-to-noise ratio and thus, a consistent error distribution. In fact, although the error distribution was not considered to be a significant factor for the experiment, it was noted that the error counts received on the BER test set were relatively uniformly distributed throughout the five minute test periods with one exception when the operator reported “bursty” errors. This held true whether the tests were made with or without modulation index adjustments. However, it should be noted that the error distribution is unknown, and there was one report of bursty errors during a BERT.

The bent-pipe experiment was conducted over two different satellites whose mission names are IRON 7837 and IRON 7506 (IRON= Inter-Range Operation Number). The two satellites are in highly elliptical orbits that bring them within view of the ground antennas in Sunnyvale, CA, at approximately the same time each day and give long view periods. The link measurement data for each spacecraft is summarized in Table 3. For IRON 7837, maximum azimuth occurred at maximum elevation, while for IRON 7506, maximum azimuth corresponded to minimum elevation. Maximum range occurred at maximum azimuth for both links. The bit error rate measurements are not summarized in the table because the high values do not necessarily correspond to experiment test runs; the BER measurements are discussed in detail below. The minimum error values were 0 errors in 5 minutes for both links. For purposes of analysis, the minimum BER was set to a value of $1x10^{-8}$. All link measurements were made on the 256 Kbps downlink.

Table 3. Link Measurement Summary Data

Parameter	IRON 7837	IRON 7506
Minimum range	17,004 nm	19,906 nm
Maximum range	22,639 nm	22,848 nm
Minimum signal strength	-113 dBm	-105 dBm
Maximum signal strength	-105 dBm	-103 dBm
Minimum antenna elevation	17 deg	32 deg
Maximum antenna elevation	36 deg	38 deg
Minimum antenna azimuth	302 deg	307 deg
Maximum antenna azimuth	317 deg	325 deg

We initially attempted to assign BER values to each run by using a log-linear interpolation between the pairs of BER measurements bracketing a series of runs (see Appendix A). Basically, the equation of the line connecting the two measurement values was determined, and then the equation was used to calculate a BER value for each run. For readers unfamiliar with this technique, refer to the example for Equation B.3-4 in Appendix B. This approach initially appeared to provide good results, but as the throughput and bit efficiency analyses progressed, we encountered a number of data correlation problems. The problems were traced back to the assigned BERs. Further examination of the BER data showed that, although it was generally reliable, there was a sufficient amount of variability in the link at certain times that the interpolated values were just not close enough to provide the precision necessary. We had to devise an alternative means for determining BER for each run.

We ultimately decided to use the number of packet retransmissions as an indicator of BER.

The error rate derived from the packet retransmission count was straightforward, but it assumes that the software was operating as intended; i.e., that a request for a retransmission was made only when needed, and that only the requested packets were sent. In other words, it was assumed that only the proper number of retransmissions were made. In order to check this assumption for validity, it was necessary to compare the derived values to the measured values.

The derived BERs were calculated as follows:

• If there are no errors in the transmission, then the total number of packets sent is just the total amount of user data to be sent (6 mbytes in this case) divided by the number of bytes in the user data part of the transmission packet (125, 250, 500, or 1000 for this experiment). Thus, with no errors, the number of packets sent would be 48,000; 24,000; 12,000; or 6,000, depending on the packet size used.

• The actual number of packets sent is calculated by subtracting the number of bytes in the control packets from the total data sent (see Table A-3), and then dividing this value by the size of a transmission packet (user data plus packet header). This yields the integer number of total packets sent. Refer to Equation B.2-4.

• Assuming that a damaged packet has experienced only one bit error, the empirical probability of a non-error, q, is obtained by dividing the ideal (error-free) value by the actual value.

• Assuming a binomial distribution for the probability density function, the bit error rate is then calculated by the following equation:

$$BER = 1 - q^{(1/N)}$$

Where N is the number of bits in the data packet
(user data + header, in bytes) x 8

As noted above, this approach cannot differentiate between packet loss due to one bit errored in the packet, and loss due to many bits in error. As a result, the estimates obtained from this method are somewhat conservative.

To validate the assumption that the retransmissions were being made correctly, error calculations were made using the data from the laboratory runs (Table A-1). Data for each packet size at each of the four error levels used in the lab was averaged, and the minimum and maximum values identified. The results of this are shown in Figure 29.

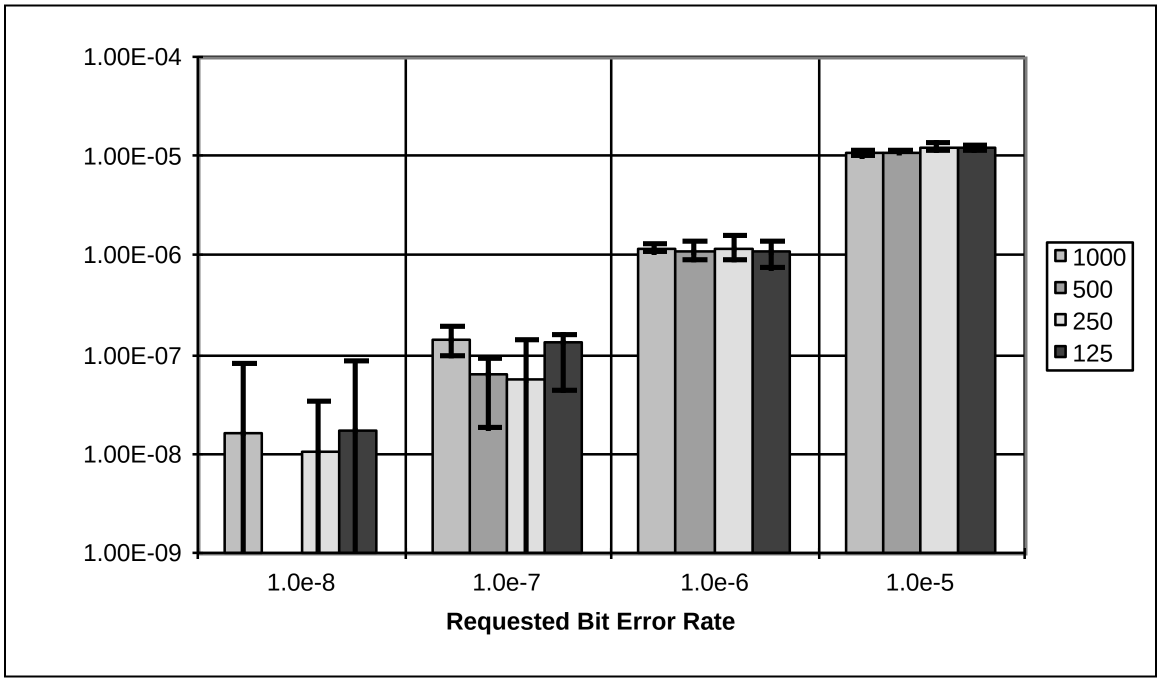


Figure 29. Bit Error Rate Derived From Laboratory Data

The chart shows that the derived error rate corresponds very well with the actual, i.e., the "requested," error rate. At first glance the range brackets for the data at the 1.0E-8 error rate seems to indicate that the correspondence is not very good. Keep in mind that at an error rate as low as this, the probability that an entire run of 6 mbytes can be made without experiencing one error is very high, and in fact, note that the entire set of runs at a packet size of 500 had no retransmissions, and therefore zero errors. As the requested error rate increases (more errors), the probability that packets will be damaged increases, and the corresponding range between the minima and maxima decreases. In summary, the average derived bit error rates correlate reasonably well with the pre-set, requested bit error rates in the lab.

The next check was to compare the error rates derived from the retransmission counts from the actual data, Table A-3, to the measured error rates described above, and shown in Table A-4. The comparisons are shown graphically in Figures 30 through 37.

The data for IRON 7837 on 5 December, shown in Figure 30, shows fairly good correlation between the measured (indicated by the triangles) and the derived values (indicated by the x's). One derived data value was much higher than the others in this series of runs, and subsequent investigation showed this to be caused by a buffer problem. This problem is discussed elsewhere.

The data for IRON 7506 on 5 December appears to be somewhat less well-correlated than the previous data set. Both the measured data and the derived data cluster around two distinct levels. An examination of the run logs revealed that this was the same period of time when the ground station personnel were reporting bursty error measurements. When this is taken into account, the data appear to correlate well.

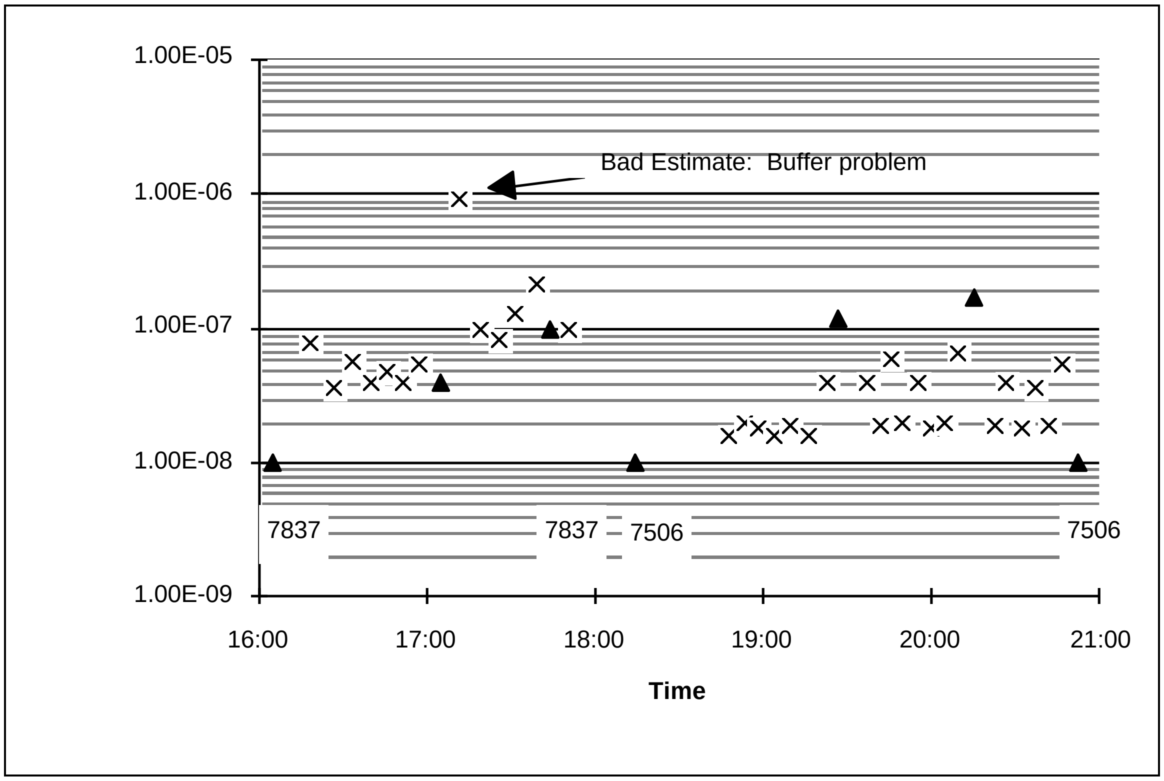


Figure 30: Derived versus Measured Errors, 5 December 1995

The data for the remaining runs, shown in Figures 31 through 37, correlate very well. The data for 7837 runs on 13 December track together, but are offset from each other, with the derived data being lower than the measured data. One possible cause of this offset is the fact that the calculation for the derived bit error rates assumes only one error per packet, and it is clear from the measured values that the link was rather noisy during these runs. Even so, the offset is not sufficient to cause concern.

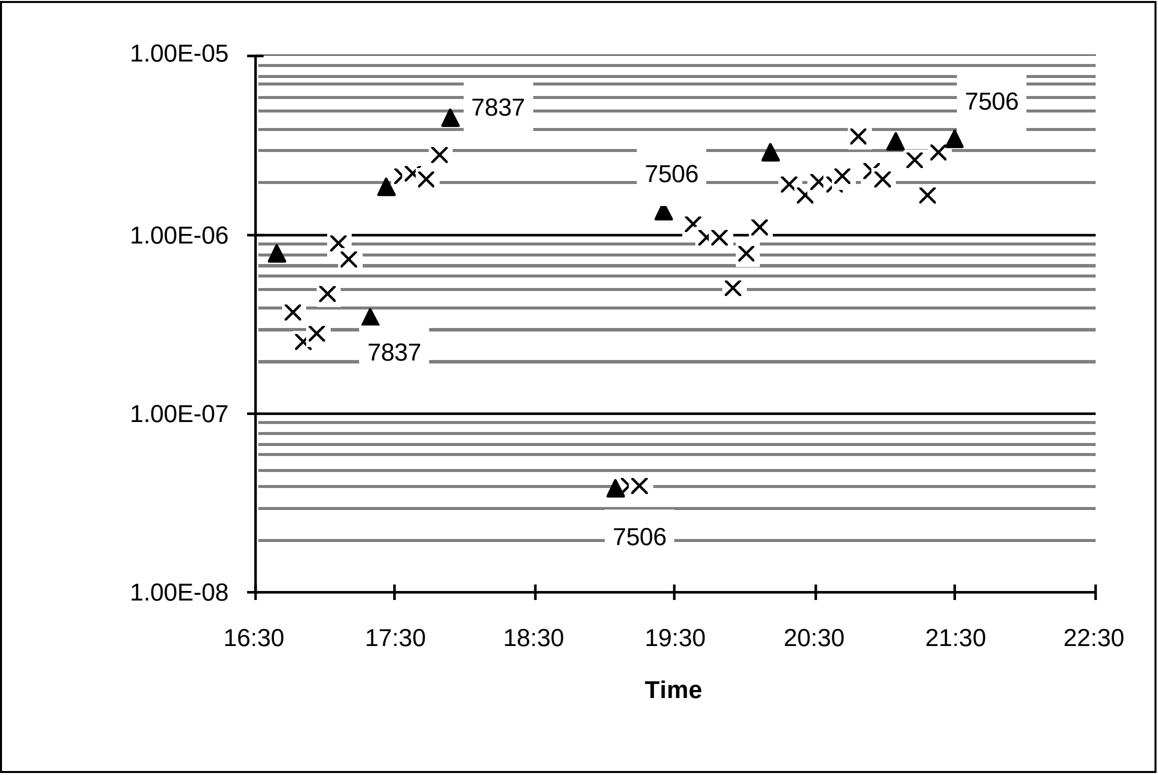


Figure 31: Derived versus Measured Errors, 6 December 1995

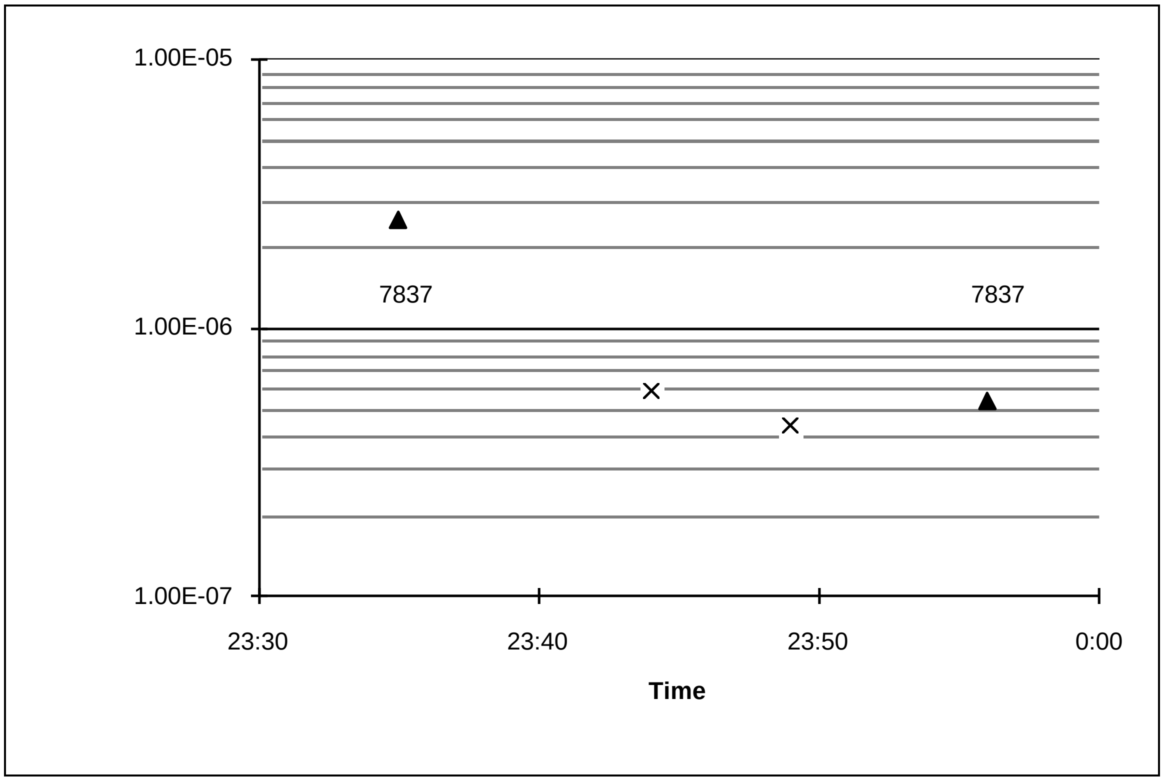


Figure 32: Derived versus Measured Errors, 7 December 1995

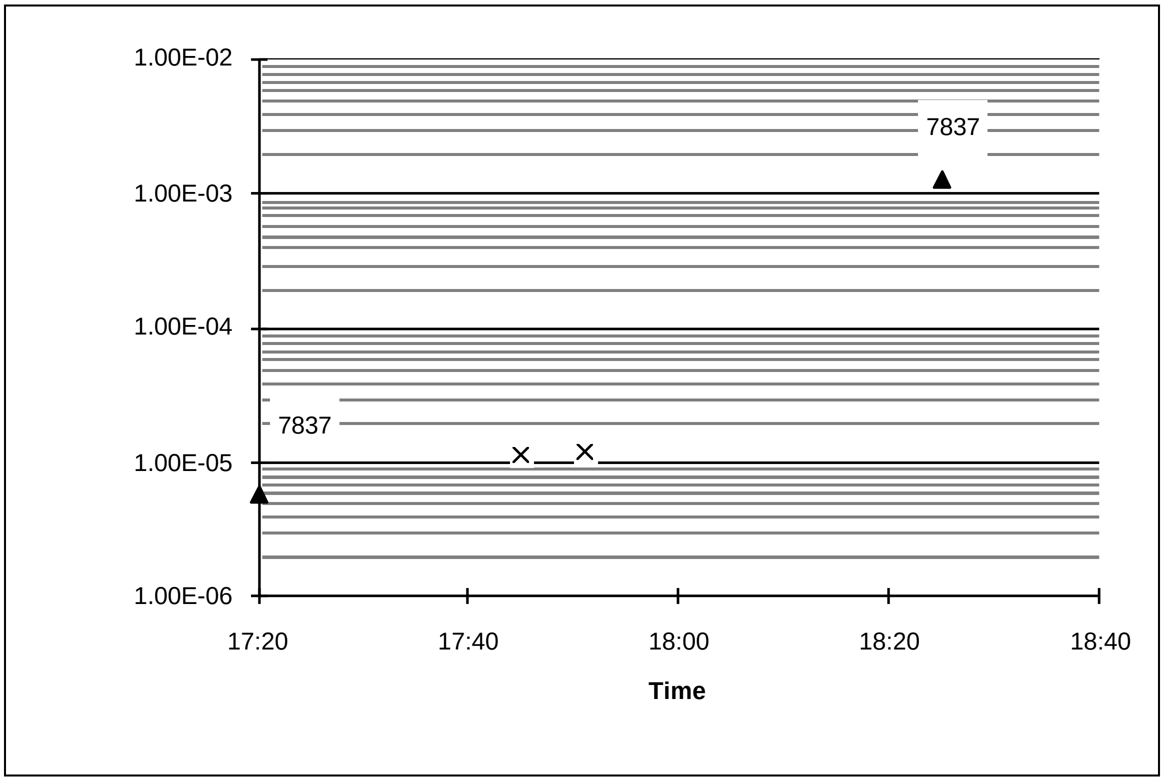


Figure 33: Derived versus Measured Errors, 8 December 1995

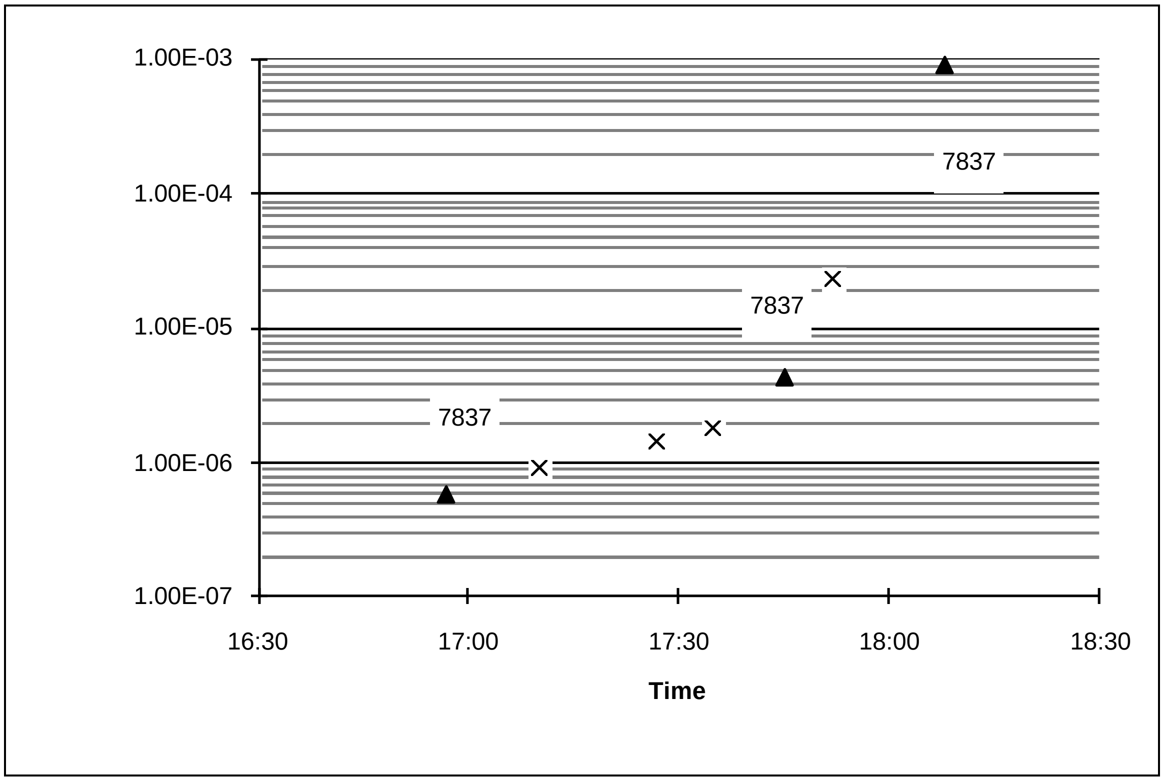


Figure 34: Derived versus Measured Errors, 12 December 1995

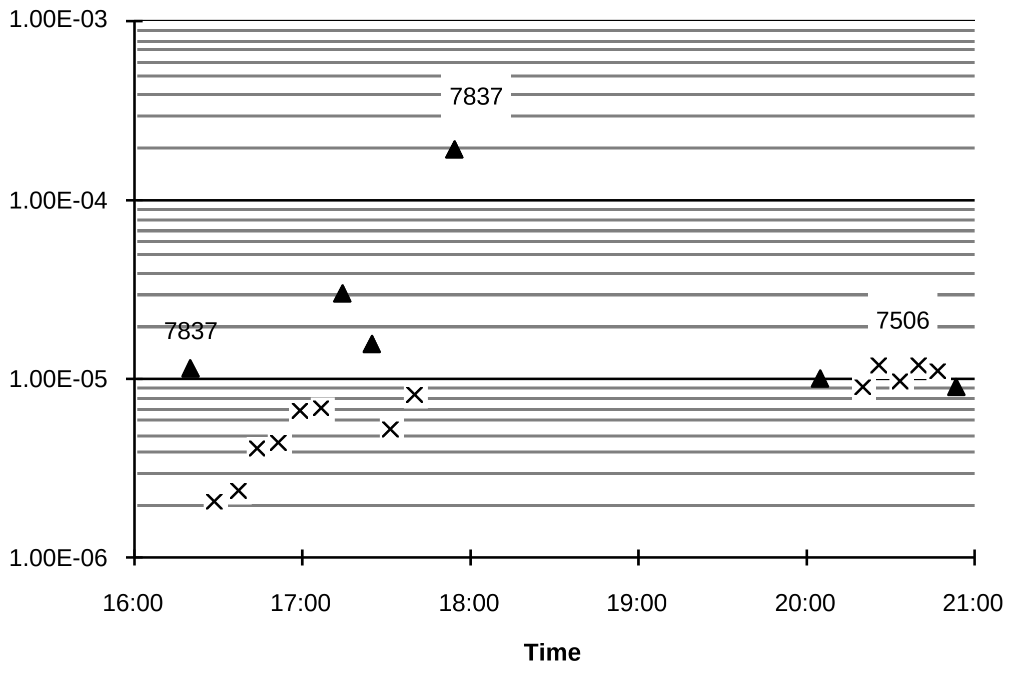


Figure 35: Derived versus Measured Errors, 13 December 1995

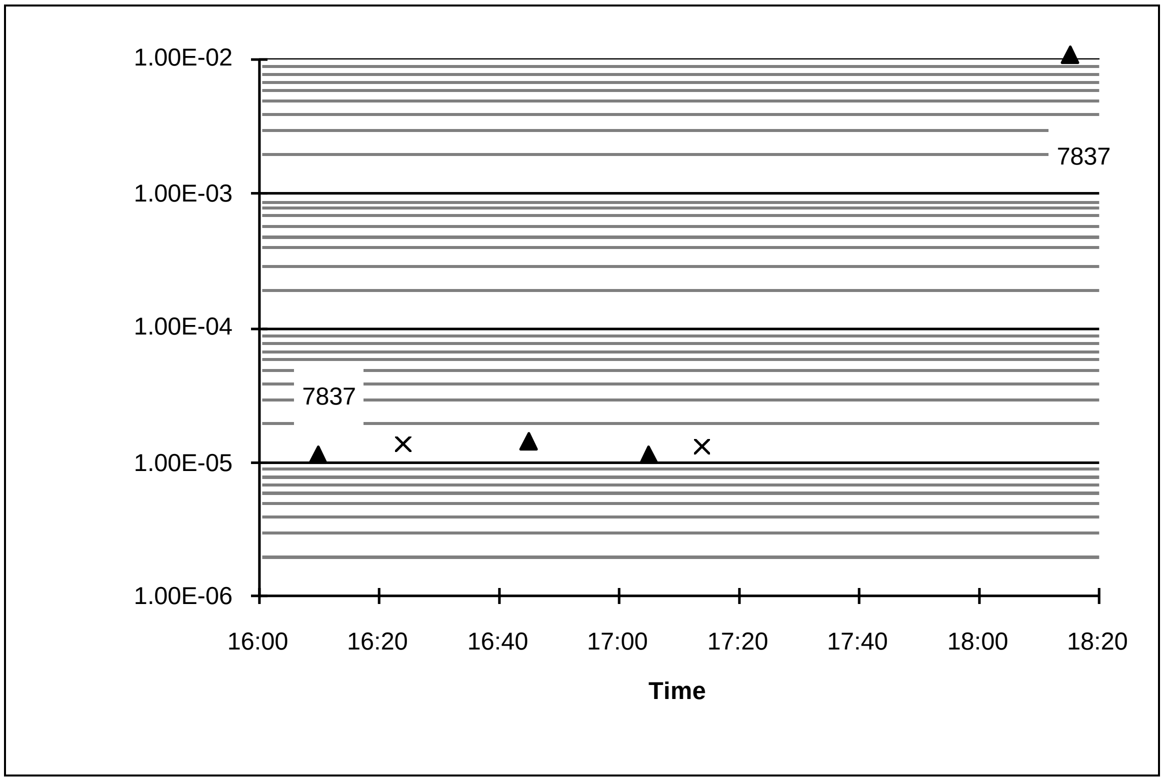


Figure 36: Derived versus Measured Errors, 14 December 1995

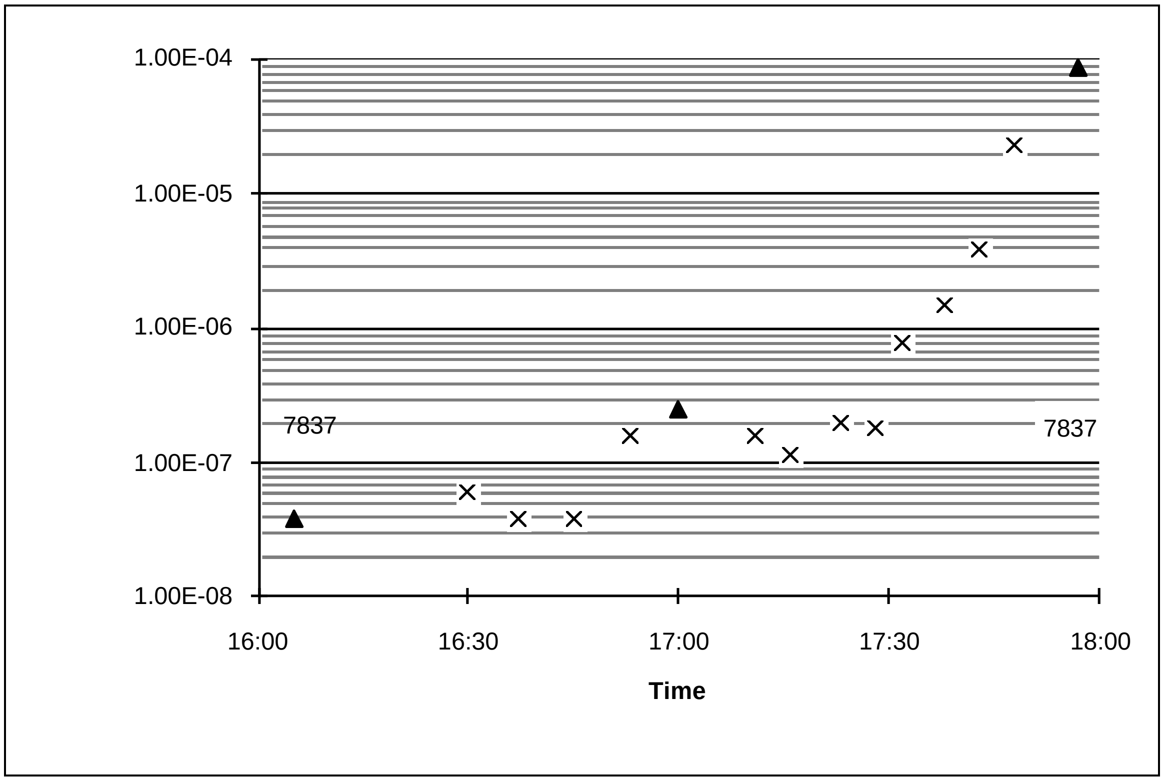
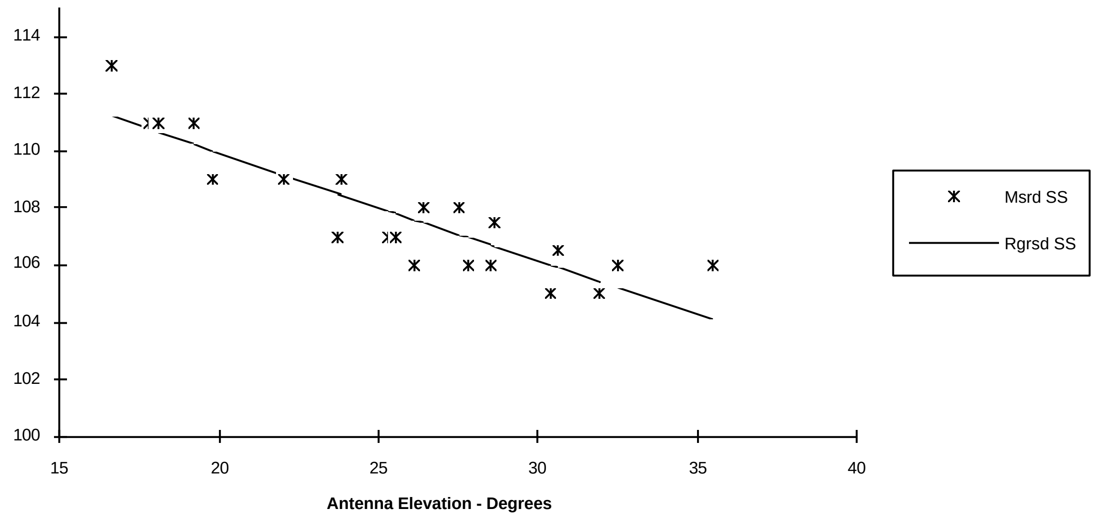


Figure 37: Derived versus Measured Errors, 15 December 1995

The forgoing analyses established the reasonableness of using the derived bit error rates for performing performance analyses on the protocol data. To summarize, the use of the derived errors provides a consistent check on the link performance because it can be calculated for each data run, whereas the measured error data provided only a snapshot of the link performance approximately every hour, with very little knowledge of the link quality in between measurements. We established confidence in the technique by first checking the lab data against the lab-generated errors, and then by correlating the bent-pipe data against the BER measurements performed on the link. The link measured data was used to interpolate errors for each run, but the uncertainty associated with this approach, coupled with the fact that some of the information on the link stability was anecdotal, suggested that the most reliable method of error estimation, in this particular case, is the derived error method.


114
112
110
108
106
104
102
100
15
20
25
30
35
40
Antenna Elevation - Degrees
Msrd SS
Rgrsd SS

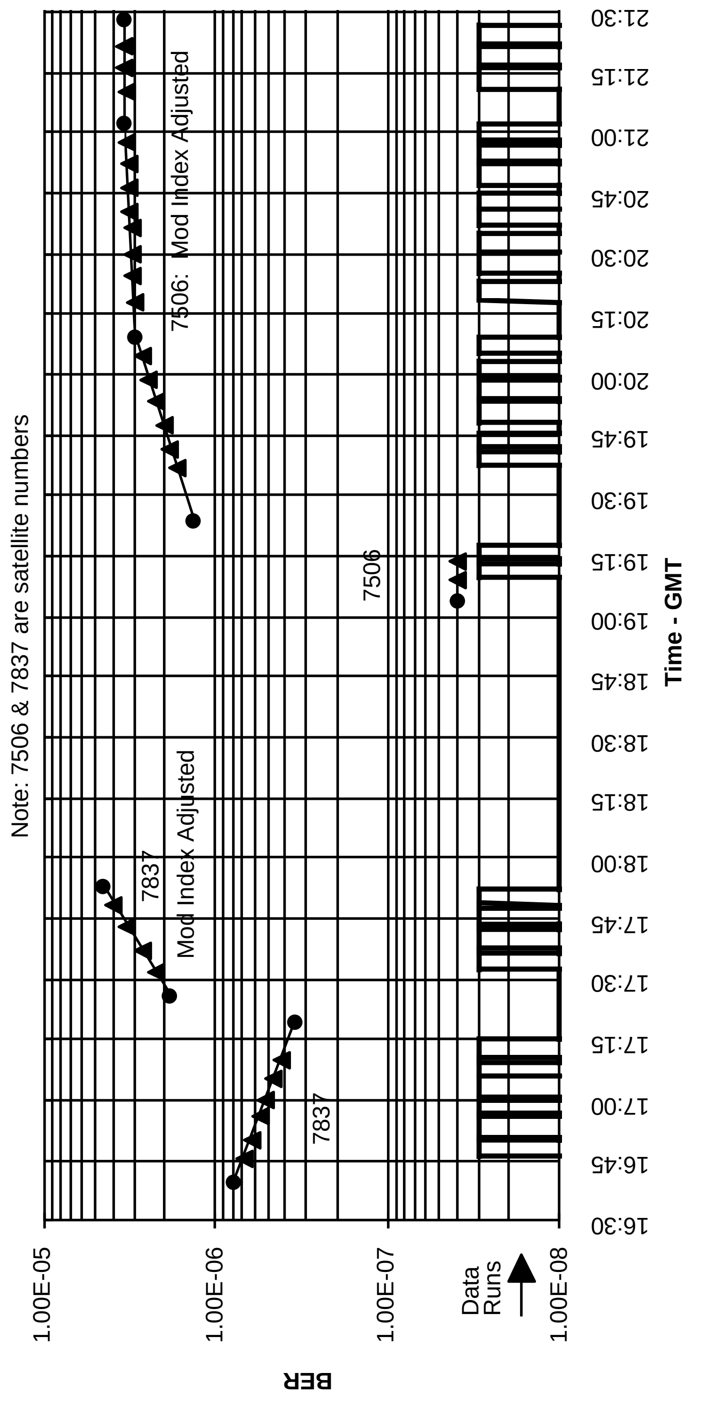


Figure 7.2-2. Assignment of BER Values to Data Runs

7.0 CONCLUSIONS AND RECOMMENDATIONS

The bent-pipe experiment provided a beneficial “first step” in SCPS protocol testing. The test provided valuable insight into the performance of SCPS-TP in a real satellite communication environment. In addition, it provided the SCPS developers with an introduction to the type of ground station equipment that may be encountered during operational use of the SCPS protocols.

The bent-pipe experiment put forth three objectives:

1. To measure the operation of SCPS-TP in a real satellite communications environment.

2. To characterize the performance of SCPS-TP.

3. To compare the performance of SCPS-TP to the Transmission Control Protocol (TCP), the standard Internet reliable transport protocol, in a satellite communications environment.

Objectives 1 and 2 were met. Objective 3 was only partially met, since the tests in support of objective 3 could only be conducted in the laboratory environment, due to restrictions of some of the interface equipment used for the tests.

We are able to draw the following conclusions from this activity:

1. SCPS-TP performed equivalently to TCP in the absence of bit errors.

2. With error rates of 10^{-7} or greater, SCPS-TP showed significantly better performance than TCP (<200 Kbps vs. 30 Kbps at $5x10^{-6}$). Limitations of the particular TCP implementation we used prevented testing in the laboratory with a broader range of packet sizes. Limitations of the test hardware in Sunnyvale prevented any testing of TCP over the actual satellite link.

3. SCPS-TP performs well over a broad range of error conditions.

    a) SCPS-TP throughput at a bit-error rate of 10^{-5} was measured at between 82% and 97% of the throughput in the lowest error cases, varying by packet size.

    b) The data channel utilization of SCPS-TP was high when the rate control parameters were properly set. In the laboratory tests, data channel utilization varied between 89% and 96%, while in the bent pipe test it varied between 70% and 96%. In analyzing the channel utilization information, we identified problems with the rate control strategy. Some of these have been repaired, while others remain under investigation.

c) The amount of acknowledgment traffic generated was low compared to the amount of traffic the acknowledgment channel could accommodate. We improved the acknowledgment strategy to further reduce the amount of acknowledgment traffic generated; the laboratory data reflects those improvements.

d) The bit-efficiency of SCPS-TP over IP for 1000-byte and 500-byte packets met the bit efficiency goal of 10% or less protocol overhead at 0 BER, which was established in the requirements phase of the activity. Further, this goal is met for 1000-byte and 500-byte packets with error rates up to approximately 5×10^{-6}.

4. The testing improved the quality of the SCPS TP implementation. We made improvements to the acknowledgment strategy and to the rate control implementation as a direct result of the bent pipe tests. We have identified areas to concentrate upon to ease deployment and field configuration of the protocol.

RECOMMENDATIONS

We offer the following recommendations as a result of the bent-pipe tests.

Broad recommendations:

1. Continue the program of SCPS testing.

   Some of the important capabilities within SCPS-TP have not yet been tested outside the laboratory - these include the Best Effort Transport Service, the User Datagram Protocol, and link outage handling. SCPS-TP should be tested in a mixed-loss environment, in which losses may be due to congestion, corruption, or link outage.

   At the time we conducted these tests, we did not have the protocols fully instrumented. While we were able to collect data about the overall characteristics of each protocol test, such as throughput and amount of data transferred, we were not able to gather detailed, instantaneous data. Subsequent protocol tests should be augmented with the ability to gather information about the instantaneous dynamics of protocol operation.

2. Expand the scope of the testing to more faithfully represent operational communication requirements. The bent-pipe experiment verified the performance of SCPS-TP operating in full reliability mode, in a two-system configuration, with no other traffic present. As testing proceeds, we should attempt to identify a limited set

of operational scenarios that represent more realistic uses of the SCPS protocols. Once these scenarios have been identified, we should define and execute a program of experimentation to test the SCPS protocols with these scenarios using as realistic a system configuration as possible.

Specific recommendations:

1. We recommend that further comparative testing be conducted with an implementation of TCP that permits specification of the maximum segment size. We also recommend that a test environment be identified in which TCP can be compared directly to SCPS-TP over actual satellite communication conditions and that comparative testing be conducted.

2. We do not currently have instrumentation and analysis tools to allow us to examine the instantaneous data rate on the acknowledgment channel. This is an important capability which we should develop, in order to ensure proper performance over links with restricted acknowledgment channels, and we recommend that it be developed.

BIBLIOGRAPHY AND LIST OF REFERENCES

1. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), November 1995, *Space Communications Protocol Standards (SCPS) Bent-Pipe Experiment Plan*, SCPS-D71.50-Y-1, Headquarters NORAD/U.S. Space Command, Peterson AFB, CO, USA

2. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), January 1996, *Report Concerning Space Communications Protocol Standards, Advanced Orbiting Systems Upper Layer Protocols: Rationale, Requirements and Application Notes* , SCPS 710.0-G-0 (Draft), DOD and NASA, Jet Propulsion Laboratory, Pasadena, CA, USA

3. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), April 1995, *Concept Paper: ADVANCED ORBITING SYSTEMS, UPPER LAYER PROTOCOLS, SCPS Network Protocol (SCPS-NP) Specification*, SCPS 713.0, DRAFT, DOD and NASA, Jet Propulsion Laboratory, Pasadena, CA, USA

4. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), April 1995, *Concept Paper: ADVANCED ORBITING SYSTEMS, UPPER LAYER PROTOCOLS, SCPS Security Protocol (SCPS-SP) Specification,* SCPS 7135.0-C-1, DRAFT, DOD and NASA, Jet Propulsion Laboratory, Pasadena, CA, USA

5. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), March 1995, *Concept Paper: ADVANCED ORBITING SYSTEMS, UPPER LAYER PROTOCOLS, SCPS Transport Protocol (SCPS-TP) Specification,* SCPS 714.0, DRAFT, DOD and NASA, Jet Propulsion Laboratory, Pasadena, CA, USA

6. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), June 1995, *Concept Paper: ADVANCED ORBITING SYSTEMS, UPPER LAYER PROTOCOLS, SCPS File Transfer Protocol (SCPS-FP) Specification,* SCPS 717.0, DRAFT, DOD and NASA, Jet Propulsion Laboratory, Pasadena, CA, USA

7. The Joint NASA/DOD Space Communications Protocol Standards Technical Working Group (SCPS-TWG), September 1993, *Recommended Development of Interoperable Data Communications Standards for Dual-Use by US Civil and Military Space Projects*, DOD and NASA, Jet Propulsion Laboratory, Pasadena, CA, USA

8. Jain, Raj, 1991, *The Art of Computer Systems Performance Analysis-Techniques for Experimental Design, Measurement, Simulation, and Modeling*, John Wiley & Sons, New York, NY

9. L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP Vegas: New Techniques for Congestion Avoidance," *Proceedings of SIGCOMM 1994*, pp. 24-35, London, U. K., October 1994.

10. Fox, R., June 1989, "TCP Big Window and Nak Options," Internet Engineering Task Force document RFC 1106.

11. [Turner, J., October 1986, "New Directions in Communications (Or Which Way to the Information Age?)," *IEEE Communications* , Vol. 24, No. 10, 8-15.

12. Stevens, W. R., 1994, *TCP/IP Illustrated, Volume 1: The Protocols*, Reading, Massachusetts: Addison-Wesley

13. Jacobson, V. and R. Braden, October 1988, "TCP Extensions for Long-Delay Paths," Internet Engineering Task Force document RFC 1072.

14. V. Jacobson, R. Braden, and D. Borman, "TCP Extensions for Long-Delay Paths," Request for Comments 1323, IETF, May 1992.

15. Connolly, T., P. Amer, P. Conrad, November 1994, "An Extension to TCP: Partial Order Service," RFC 1693.

16. Cooper, G., March 1986, TCP implementation, IMAGEN Corporation.

17. IETF, 1989, "Requirements for Internet Hosts - Communications Layers," Internet Engineering Task Force document RFC 1122.

APPENDIX A: DATA

A.1 PROTOCOL DATA

Table A-1 lists the complete record for each data run recorded during the laboratory portion of the experiment. Each record consists of four parts: a set of initial conditions; the data from the initiator software; the data from the responder software; and a set of data computed from the measured data. In interpreting the data, bear in mind that in every instance, the amount of data to be transferred (the "user data") was 6,000,000 bytes.

The table has been rotated for ease in incorporating it into the report. The "headings" are contained in the first column, and each succeeding column is a data record. The headings have the following meanings:

Initial Conditions

Pkt Size: The size in bytes of the data packets.

Sel BER: The selected bit error rate. Used by Spanner for generating random errors during run.

Ack Floor: Minimum period between acknowledgments, in milliseconds.

Initiator Data

Total Data Sent: The total amount of data sent by the initiator. This includes the 6 mbytes of user data, the overhead data, and retransmission data.

Init Elpsd Sec: The elapsed time in seconds of the transmission, as measured at the initiator.

Thruput Kbps: The effective user data throughput in Kbps of the transmission.

Init rttbest: The best round trip transmission time, in seconds, as measured at the initiator.

Total Ack Data: The total quantity of acknowledgement data received at the initiator.

Responder Data

Resp rttbest: The best round trip transmission time, in seconds, as measured at the responder.

Dlyd Acks Rqustd: Delayed acknowledgments requested.

Dlys Acks Sent: Delayed acknowledgments actually sent.

Computed Data

Est Data Pkts: The estimated number of total data packets sent. See Equation B.2-4.

Chan Util Kbps: The effective channel utilization rate, in Kbps. Refer to Equation B.2-2.

Est BER: The estimated BER for the data run. Refer to the text and section A.2 for a description of the method of assigning the BER.

Table A-1. Laboratory Data

Pkt Size	1000	1000	1000	1000	1000	1000	1000	1000	1000
Sel BER	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-07	1.00E-07	1.00E-07	1.00E-07
Ack Floor	500	500	500	500	500	500	500	500	500
Total Data Sent	6313224	6313224	6313224	6313224	6312172	6319536	6323744	6316380	6324796
Init Elpsd Sec	206.15	205.94	205.94	206.03	206.01	206.21	206.31	206.00	206.35
Thruput Kbps	232.8	233.1	233.1	233.0	233.0	232.8	232.7	233.0	232.6
Init rttbest	2.10	2.16	2.16	2.15	2.04	2.16	2.16	2.13	2.16
Total Ack Data	21292	21240	21188	21240	21128	21652	21892	21308	21780
Resp rttbest	0.52	0.52	0.56	0.56	0.56	0.52	0.52	0.56	0.56
Dlyd Acks Rqustd	405	404	403	404	403	404	404	403	403
Dlyd Acks Sent	406	405	404	405	404	405	405	404	404
Est Data Pkts	6001	6001	6001	6001	6000	6007	6011	6004	6012
Chan Util Kbps	245.0	245.2	245.2	245.1	245.1	245.2	245.2	245.3	245.2
Est BER	1.98E-08	1.98E-08	1.98E-08	1.98E-08	1.00E-12	1.39E-07	2.18E-07	7.92E-08	2.37E-07

Table A-1. Laboratory Data (continued)

Pkt Size	1000	1000	1000	1000	1000	1000	1000	1000	1000
Sel BER	1.00E-07	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-05	1.00E-05	1.00E-05
Ack Floor	500	500	500	500	500	500	500	500	500
Total Data Sent	6313224	6374240	6378500	6357460	6362668	6379552	6867680	6897136	6886616
Init Elpsd Sec	206.12	209.58	211.26	208.72	207.73	209.58	227.15	228.54	230.36
Thruput Kbps	232.9	229.0	227.2	230.0	231.1	229.0	211.3	210.0	208.4
Init rttbest	2.17	1.86	0.55	0.50	1.95	0.53	0.53	0.54	0.50
Total Ack Data	21292	26580	25016	23728	23984	24868	41224	44908	47404
Resp rttbest	0.52	0.56	0.52	0.50	0.52	0.52	0.54	0.53	0.52
Dlyd Acks Rqustd	405	416	412	408	406	409	443	439	444
Dlyd Acks Sent	406	452	413	409	407	410	445	498	566
Est Data Pkts	6001	6059	6063	6043	6048	6064	6528	6556	6546
Chan Util Kbps	245.0	243.3	241.5	243.7	245.0	243.5	241.9	241.4	239.2
Est BER	1.98E-08	1.16E-06	1.24E-06	8.49E-07	9.47E-07	1.26E-06	1.00E-05	1.05E-05	1.03E-05
Pkt Size	1000	1000	1000	1000	1000	1000	1000	1000	1000
Sel BER	1.00E-05	1.00E-05	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-07	1.00E-07
Ack Floor	500	500	250	250	250	250	250	250	250
Total Data Sent	6873992	6893980	6312172	6316432	6312172	6312292	6312172	6317432	6317432
Init Elpsd Sec	226.28	228.62	205.99	207.23	206.33	206.10	205.91	206.21	206.20
Thruput Kbps	212.1	210.0	233.0	231.6	232.6	232.9	233.1	232.8	232.8
Init rttbest	0.54	0.53	2.17	0.51	1.92	2.10	2.10	2.04	1.80
Total Ack Data	41292	42628	41512	41856	41460	41528	41408	41760	41760
Resp rttbest	0.53	0.54	0.56	0.51	0.57	0.56	0.52	0.52	0.56
Dlyd Acks Rqustd	442	443	795	797	794	795	793	794	794
Dlyd Acks Sent	449	453	796	798	795	795	794	795	795
Est Data Pkts	6534	6553	6000	6004	6000	6000	6000	6005	6005
Chan Util Kbps	243.0	241.2	245.1	243.8	244.7	245.0	245.2	245.1	245.1
Est BER	1.01E-05	1.05E-05	1.00E-12	8.02E-08	1.00E-12	2.26E-09	1.00E-12	9.90E-08	9.90E-08

Table A-1. Laboratory Data (continued)

Pkt Size	1000	1000	1000	1000	1000	1000	1000	1000	1000
Sel BER	1.00E-07	1.00E-07	1.00E-07	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-05
Ack Floor	250	250	250	250	250	250	250	250	250
Total Data Sent	6321640	6318484	6322692	6375344	6382656	6375396	6368980	6371136	6936060
Init Elpsd Sec	206.65	206.59	206.68	211.18	208.57	209.82	208.21	208.81	229.59
Thruput Kbps	232.3	232.3	232.2	227.3	230.1	228.8	230.5	229.9	209.1
Init rttbest	1.93	1.82	2.05	0.50	1.94	0.50	2.03	0.55	0.51
Total Ack Data	42052	41820	42112	45020	45580	45460	44824	44980	69408
Resp rttbest	0.58	0.52	0.52	0.56	0.56	0.50	0.52	0.52	0.50
Dlyd Acks Rqustd	794	794	795	802	804	806	801	805	872
Dlyd Acks Sent	795	795	796	803	805	807	802	806	978
Est Data Pkts	6009	6006	6010	6060	6067	6060	6054	6056	6593
Chan Util Kbps	244.7	244.7	244.7	241.5	244.8	243.1	244.7	244.1	241.7
Est BER	1.78E-07	1.19E-07	1.98E-07	1.18E-06	1.32E-06	1.18E-06	1.06E-06	1.10E-06	1.12E-05

Pkt Size	1000	1000	1000	1000	500	500	500	500	500
Sel BER	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-08
Ack Floor	250	250	250	250	500	500	500	500	500
Total Data Sent	6890824	6929748	6967620	6880304	6624172	6624172	6624724	6625828	6624172
Init Elpsd Sec	230.01	231.17	234.02	227.23	216.90	216.89	216.93	216.91	217.05
Thruput Kbps	208.7	207.6	205.1	211.2	221.3	221.3	221.3	221.3	221.1
Init rttbest	0.53	0.55	0.50	0.55	2.60	2.85	2.82	2.92	2.62
Total Ack Data	71064	70180	89080	64816	22220	22272	22228	22452	22272
Resp rttbest	0.53	0.52	0.50	0.52	0.56	0.52	0.56	0.56	0.56
Dlyd Acks Rqustd	872	873	875	865	424	424	423	425	425
Dlyd Acks Sent	1042	1071	1382	866	425	425	424	426	426
Est Data Pkts	6550	6587	6623	6540	12000	12000	12001	12003	12000
Chan Util Kbps	239.7	239.8	238.2	242.2	244.3	244.3	244.3	244.4	244.1
Est BER	1.04E-05	1.11E-05	1.17E-05	1.02E-05	1.00E-12	1.00E-12	1.89E-08	5.66E-08	1.00E-12

Table A-1. Laboratory Data (continued)

Pkt Size	500	500	500	500	500	500	500	500	500
Sel BER	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-06	1.00E-06	1.00E-06	1.00E-06
Ack Floor	500	500	500	500	500	500	500	500	500
Total Data Sent	6625828	6626380	6626932	6625276	6628588	6667832	6656240	6660656	6656792
Init Elpsd Sec	216.95	216.96	217.00	216.95	217.01	221.07	222.51	220.08	220.11
Thruput Kbps	221.2	221.2	221.2	221.2	221.2	217.1	215.7	218.1	218.1
Init rttbest	2.66	2.73	2.91	2.59	2.76	0.50	0.54	0.52	0.50
Total Ack Data	22452	22616	22572	22392	22804	26560	25496	25984	25616
Resp rttbest	0.56	0.52	0.57	0.52	0.57	0.56	0.52	0.52	0.52
Dlyd Acks Rqustd	424	426	425	425	425	429	427	427	429
Dlyd Acks Sent	425	427	426	426	426	431	428	428	431
Est Data Pkts	12003	12004	12005	12002	12008	12079	12058	12066	12059
Chan Util Kbps	244.3	244.3	244.3	244.3	244.4	241.3	239.3	242.1	241.9
Est BER	5.66E-08	7.55E-08	9.43E-08	3.77E-08	1.51E-07	1.49E-06	1.09E-06	1.24E-06	1.11E-06
Pkt Size	500	500	500	500	500	500	500	500	500
Sel BER	1.00E-06	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-08	1.00E-08	1.00E-08
Ack Floor	500	500	500	500	500	500	250	250	250
Total Data Sent	6645752	6951560	6983024	6939968	6942728	6926168	6624172	6624172	6624172
Init Elpsd Sec	220.12	230.47	232.16	230.50	231.96	229.41	217.36	216.98	216.87
Thruput Kbps	218.1	208.3	206.8	208.2	206.9	209.2	220.8	221.2	221.3
Init rttbest	0.55	0.50	0.52	0.50	0.54	0.52	2.43	2.80	2.97
Total Ack Data	24528	51788	52948	48316	49576	49444	43592	43592	43592
Resp rttbest	0.52	0.56	0.51	0.52	0.52	0.52	0.52	0.56	0.56
Dlyd Acks Rqustd	427	455	459	449	454	451	835	835	835
Dlyd Acks Sent	428	456	460	450	455	452	836	835	836
Est Data Pkts	12039	12593	12650	12572	12577	12547	12000	12000	12000
Chan Util Kbps	241.5	241.3	240.6	240.9	239.5	241.5	243.8	244.2	244.4
Est BER	7.37E-07	1.09E-05	1.19E-05	1.05E-05	1.06E-05	1.01E-05	1.00E-12	1.00E-12	1.00E-12

Table A-1. Laboratory Data (continued)

Pkt Size	500	500	500	500	500	500	500	500	500
Sel BER	1.00E-08	1.00E-08	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-06	1.00E-06
Ack Floor	250	250	250	250	250	250	250	250	250
Total Data Sent	6624172	6624172	6626932	6626380	6626380	6624724	6625828	6663968	6664468
Init Elpsd Sec	217.02	216.94	217.20	216.96	217.09	216.86	216.87	219.56	218.40
Thruput Kbps	221.2	221.3	221.0	221.2	221.1	221.3	221.3	218.6	219.8
Init rttbest	2.73	2.79	2.80	2.59	2.71	2.90	2.74	0.50	2.40
Total Ack Data	43644	43696	44048	43780	43832	43652	43720	47984	47992
Resp rttbest	0.52	0.56	0.56	0.52	0.52	0.56	0.52	0.50	0.56
Dlyd Acks Rqustd	836	837	838	834	835	835	834	842	840
Dlyd Acks Sent	837	838	839	835	836	836	835	843	841
Est Data Pkts	12000	12000	12005	12004	12004	12001	12003	12072	12073
Chan Util Kbps	244.2	244.3	244.1	244.3	244.2	244.4	244.4	242.8	244.1
Est BER	1.00E-12	1.00E-12	9.43E-08	7.55E-08	7.55E-08	1.89E-08	5.66E-08	1.36E-06	1.37E-06
Pkt Size	500	500	500	500	500	500	500	500	250
Sel BER	1.00E-06	1.00E-06	1.00E-06	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-08
Ack Floor	250	250	250	250	250	250	250	250	500
Total Data Sent	6652928	6651220	6652928	6944936	6977504	6939916	6957632	6958736	7249078
Init Elpsd Sec	221.48	217.87	221.66	232.20	231.70	228.84	231.39	232.37	242.81
Thruput Kbps	216.7	220.3	216.5	206.7	207.2	209.8	207.4	206.6	197.7
Init rttbest	0.50	2.62	0.55	0.52	0.50	1.21	0.52	0.52	5.12
Total Ack Data	46724	46388	46852	72712	74112	72208	71392	73952	25052
Resp rttbest	0.56	0.56	0.52	0.52	0.50	0.56	0.52	0.52	0.57
Dlyd Acks Rqustd	841	838	839	886	891	878	887	883	474
Dlyd Acks Sent	842	839	840	887	892	879	888	884	475
Est Data Pkts	12052	12049	12052	12581	12640	12572	12604	12606	24003
Chan Util Kbps	240.3	244.2	240.1	239.3	240.9	242.6	240.5	239.6	238.8
Est BER	9.81E-07	9.23E-07	9.81E-07	1.07E-05	1.18E-05	1.05E-05	1.11E-05	1.12E-05	5.17E-08

Table A-1. Laboratory Data (continued)

Pkt Size	250	250	250	250	250	250	250	250	250
Sel BER	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-07
Ack Floor	500	500	500	500	500	500	500	500	500
Total Data Sent	7248292	7248172	7249146	7248172	7249380	7249682	7251848	7248172	7252218
Init Elpsd Sec	242.71	242.71	242.82	242.75	242.93	242.79	250.84	242.78	243.00
Thruput Kbps	197.8	197.8	197.7	197.7	197.6	197.7	191.4	197.7	197.5
Init rttbest	5.63	5.66	5.63	5.65	5.33	5.65	0.50	5.66	4.61
Total Ack Data	24732	24768	25068	24716	25008	25120	25524	24768	25504
Resp rttbest	0.56	0.56	0.56	0.56	0.56	0.57	0.56	0.56	0.56
Dlyd Acks Rqustd	471	473	473	472	473	472	475	472	472
Dlyd Acks Sent	472	474	474	473	474	475	478	474	473
Est Data Pkts	24000	24000	24003	24000	24004	24005	24012	24000	24013
Chan Util Kbps	238.9	238.9	238.8	238.9	238.7	238.9	231.3	238.8	238.8
Est BER	6.85E-09	1.00E-12	5.56E-08	1.00E-12	6.90E-08	8.62E-08	2.10E-07	1.00E-12	2.31E-07
Pkt Size	250	250	250	250	250	250	250	250	250
Sel BER	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-05	1.00E-05	1.00E-05	1.00E-05
Ack Floor	500	500	500	500	500	500	500	500	500
Total Data Sent	7275706	7275050	7269786	7258492	7266460	7439692	7446034	7429492	7434860
Init Elpsd Sec	245.87	243.71	247.44	247.56	250.89	253.85	252.96	252.20	250.93
Thruput Kbps	195.2	197.0	194.0	193.9	191.3	189.1	189.8	190.3	191.3
Init rttbest	0.55	5.49	0.55	0.54	0.55	0.50	0.50	0.51	0.50
Total Ack Data	29732	29484	28960	26904	28276	56256	57668	55888	61380
Resp rttbest	0.52	0.55	0.52	0.51	0.52	0.51	0.51	0.51	0.50
Dlyd Acks Rqustd	476	475	476	473	474	490	489	485	490
Dlyd Acks Sent	477	477	478	477	478	491	503	518	581
Est Data Pkts	24091	24089	24072	24034	24061	24634	24655	24600	24618
Chan Util Kbps	236.7	238.8	235.0	234.6	231.7	234.5	235.5	235.7	237.0
Est BER	1.57E-06	1.53E-06	1.23E-06	5.89E-07	1.04E-06	1.08E-05	1.11E-05	1.02E-05	1.05E-05

Table A-1. Laboratory Data (continued)

Pkt Size	250	250	250	250	250	250	250	250	250
Sel BER	1.00E-05	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-07	1.00E-07	1.00E-07
Ack Floor	500	250	250	250	250	250	250	250	250
Total Data Sent	7456422	7248776	7248172	7248712	7248172	7248172	7249682	7248172	7248776
Init Elpsd Sec	252.21	242.77	242.74	242.77	242.68	242.72	242.73	242.68	242.82
Thruput Kbps	190.3	197.7	197.7	197.7	197.8	197.8	197.8	197.8	197.7
Init rttbest	0.50	5.87	5.85	5.83	5.85	5.87	5.86	5.83	5.87
Total Ack Data	56512	49068	48740	48844	48688	48844	49248	48844	48964
Resp rttbest	0.56	0.56	0.52	0.56	0.52	0.56	0.57	0.52	0.52
Dlyd Acks Rqustd	487	937	934	936	933	937	937	936	936
Dlyd Acks Sent	489	938	935	937	934	937	939	937	937
Est Data Pkts	24690	24002	24000	24002	24000	24000	24005	24000	24002
Chan Util Kbps	236.5	238.9	238.9	238.9	238.9	238.9	238.9	238.9	238.8
Est BER	1.17E-05	3.45E-08	1.00E-12	3.08E-08	1.00E-12	1.00E-12	8.62E-08	1.00E-12	3.45E-08
Pkt Size	250	250	250	250	250	250	250	250	250
Sel BER	1.00E-07	1.00E-07	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-05	1.00E-05
Ack Floor	250	250	250	250	250	250	250	250	250
Total Data Sent	7250588	7248474	7263928	7271176	7276612	7267250	7266594	7447846	7488918
Init Elpsd Sec	242.82	242.71	246.27	251.54	244.75	248.00	243.39	251.94	252.28
Thruput Kbps	197.7	197.8	194.9	190.8	196.1	193.5	197.2	190.5	190.3
Init rttbest	5.85	5.81	0.50	0.50	0.50	0.50	5.63	0.51	0.51
Total Ack Data	49160	48800	51992	53184	54100	52272	52444	79748	92256
Resp rttbest	0.57	0.52	0.52	0.56	0.50	0.52	0.56	0.52	0.52
Dlyd Acks Rqustd	933	934	933	941	939	934	935	956	965
Dlyd Acks Sent	935	935	940	943	941	937	937	960	1064
Est Data Pkts	24008	24001	24052	24076	24094	24063	24061	24661	24797
Chan Util Kbps	238.9	238.9	236.0	231.3	237.8	234.4	238.8	236.5	237.5
Est BER	1.38E-07	1.72E-08	8.99E-07	1.31E-06	1.62E-06	1.09E-06	1.05E-06	1.12E-05	1.35E-05

Table A-1. Laboratory Data (continued)

Pkt Size	250	250	250	125	125	125	125	125	125
Sel BER	1.00E-05	1.00E-05	1.00E-05	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-08	1.00E-07
Ack Floor	250	250	250	500	500	500	500	500	500
Total Data Sent	7459926	7462342	7454188	8496172	8497286	8496172	8496172	8497411	8497286
Init Elpsd Sec	252.31	251.80	252.21	296.39	305.79	296.37	296.40	296.42	305.49
Thruput Kbps	190.2	190.6	190.3	162.0	157.0	162.0	161.9	161.9	157.1
Init rttbest	0.54	0.50	0.52	7.07	0.50	7.05	7.07	6.90	0.55
Total Ack Data	82984	84512	81672	30072	30476	30072	30072	30132	30476
Resp rttbest	0.52	0.50	0.51	0.52	0.52	0.56	0.56	0.52	0.56
Dlyd Acks Rqustd	962	964	961	575	575	575	575	574	575
Dlyd Acks Sent	1022	1022	994	576	578	576	576	576	578
Est Data Pkts	24701	24709	24682	48000	48006	48000	48000	48007	48006
Chan Util Kbps	236.5	237.1	236.4	229.3	222.3	229.3	229.3	229.3	222.5
Est BER	1.19E-05	1.21E-05	1.16E-05	1.00E-12	9.26E-08	1.00E-12	1.00E-12	1.03E-07	9.26E-08
Pkt Size	125	125	125	125	125	125	125	125	125
Sel BER	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-06	1.00E-06	1.00E-06	1.00E-06	1.00E-06
Ack Floor	500	500	500	500	500	500	500	500	500
Total Data Sent	8498171	8496172	8498827	8497057	8513924	8512508	8512810	8504897	8510561
Init Elpsd Sec	298.34	296.43	296.45	296.43	298.83	299.59	297.00	298.98	298.56
Thruput Kbps	160.9	161.9	161.9	161.9	160.6	160.2	161.6	160.5	160.8
Init rttbest	0.50	7.02	6.75	6.26	0.52	0.50	3.02	0.55	0.50
Total Ack Data	30888	30072	31024	30372	35800	35328	35244	33108	34668
Resp rttbest	0.51	0.56	0.56	0.52	0.55	0.52	0.56	0.56	0.52
Dlyd Acks Rqustd	576	575	568	573	575	573	574	570	571
Dlyd Acks Sent	579	576	577	576	581	580	581	578	579
Est Data Pkts	48011	48000	48015	48005	48100	48092	48094	48049	48081
Chan Util Kbps	227.9	229.3	229.3	229.3	227.9	227.3	229.3	227.6	228.0
Est BER	1.66E-07	1.00E-12	2.21E-07	7.36E-08	1.47E-06	1.36E-06	1.38E-06	7.25E-07	1.20E-06

Table A-1. Laboratory Data (continued)

Pkt Size	125	125	125	125	125	125	125	125	125
Sel BER	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-08	1.00E-08	1.00E-08	1.00E-08
Ack Floor	500	500	500	500	500	250	250	250	250
Total Data Sent	8638709	8666498	8654462	8638001	8649204	8496172	8496172	8497234	8496172
Init Elpsd Sec	304.69	306.77	306.10	303.11	304.28	296.40	296.44	296.43	296.37
Thruput Kbps	157.5	156.5	156.8	158.4	157.8	161.9	161.9	161.9	162.0
Init rttbest	0.50	0.50	0.50	0.51	0.55	7.21	7.18	6.96	14.43
Total Ack Data	74192	89980	84904	84980	80644	59660	59556	59752	59452
Resp rttbest	0.51	0.51	0.51	0.51	0.52	0.52	0.52	0.56	0.56
Dlyd Acks Rqustd	585	584	587	581	590	1144	1142	1138	1140
Dlyd Acks Sent	748	892	786	929	805	1145	1143	1141	1141
Est Data Pkts	48805	48962	48894	48801	48865	48000	48000	48006	48000
Chan Util Kbps	226.8	226.0	226.2	228.0	227.4	229.3	229.3	229.3	229.3
Est BER	1.17E-05	1.40E-05	1.30E-05	1.17E-05	1.26E-05	1.00E-12	1.00E-12	8.83E-08	1.00E-12
Pkt Size	125	125	125	125	125	125	125	125	125
Sel BER	1.00E-08	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-07	1.00E-06	1.00E-06	1.00E-06
Ack Floor	250	250	250	250	250	250	250	250	250
Total Data Sent	8496172	8497942	8498119	8496703	8497942	8497994	8512810	8505376	8508437
Init Elpsd Sec	296.43	296.42	296.45	296.41	296.44	299.14	296.96	296.71	304.07
Thruput Kbps	161.9	161.9	161.9	161.9	161.9	160.5	161.6	161.8	157.9
Init rttbest	7.18	7.19	7.18	7.14	5.54	0.50	7.00	6.99	0.55
Total Ack Data	59660	60096	60156	59720	60096	59804	64744	62792	63452
Resp rttbest	0.56	0.56	0.56	0.52	0.56	0.55	0.52	0.56	0.52
Dlyd Acks Rqustd	1144	1137	1138	1144	1138	1140	1139	1135	1142
Dlyd Acks Sent	1145	1143	1143	1145	1143	1142	1147	1151	1150
Est Data Pkts	48000	48010	48011	48003	48010	48010	48094	48052	48069
Chan Util Kbps	229.3	229.3	229.3	229.3	229.3	227.3	229.3	229.3	223.9
Est BER	1.00E-12	1.47E-07	1.62E-07	4.41E-08	1.47E-07	1.51E-07	1.38E-06	7.65E-07	1.02E-06

Table A-1. Laboratory Data (concluded)

Pkt Size	125	125	125	125	125	125	125
Sel BER	1.00E-06	1.00E-06	1.00E-05	1.00E-05	1.00E-05	1.00E-05	1.00E-05
Ack Floor	250	250	250	250	250	250	250
Total Data Sent	8508083	8512154	8644550	8648033	8652869	8637293	8641718
Init Elpsd Sec	298.71	297.99	304.67	304.14	304.28	303.58	303.66
Thruput Kbps	160.7	161.1	157.5	157.8	157.8	158.1	158.1
Init rttbest	0.51	0.50	0.51	0.51	0.50	0.50	0.50
Total Ack Data	63580	64176	104036	104376	111592	106788	105212
Resp rttbest	0.51	0.51	0.52	0.52	0.56	0.51	0.51
Dlyd Acks Rqustd	1141	1144	1163	1162	1157	1158	1158
Dlyd Acks Sent	1152	1146	1294	1285	1384	1353	1320
Est Data Pkts	48067	48090	48838	48858	48885	48797	48822
Chan Util Kbps	227.9	228.5	227.0	227.5	227.5	227.6	227.7
Est BER	9.89E-07	1.33E-06	1.22E-05	1.25E-05	1.29E-05	1.16E-05	1.20E-05

Table A-2 lists the complete record for each TCP data run recorded during the laboratory portion of the experiment.. In interpreting the data, bear in mind that in every instance, the amount of data to be transferred (the “user data”) was 6,000,000 bytes, and the size of a data packet was 512 bytes in all cases.

The headings have the following meanings:

Error Rate: The selected bit error rate. Used by Spanner for generating random errors during run.

Elapsed Time: The elapsed time of the transmission, in seconds.

Throughput: The effective user data throughput in Kbps of the transmission.

Table A-2. TCP Data

Error Rate	Elapsed Time	Throughput
1.00E-08	205.94	227.6
1.00E-08	206.06	227.5
1.00E-08	206.13	227.4
1.00E-08	214.39	218.6
1.00E-08	221.10	212.0
1.00E-07	205.99	227.5
1.00E-07	239.12	196.0
1.00E-07	247.96	189.0
1.00E-07	280.87	166.9
1.00E-07	287.78	162.9
1.00E-06	498.76	94.0
1.00E-06	568.98	82.4
1.00E-06	573.59	81.8
1.00E-06	628.66	74.6
1.00E-06	685.84	68.3
5.00E-06	1553.28	30.9
5.00E-06	1527.73	30.7
5.00E-06	1588.00	29.5

Table A-3. M-22 Experiment Data

IRON	7837	7837	7837	7837	7837	7837	7837	7837	7837
Date	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/5
Run No.	1	2	3	4	5	6	7	8	9
Pkt Size	1000	250	500	1000	125	500	250	250	125
Start Time	16:19	16:27	16:34	16:40	16:46	16:52	16:58	17:12	17:19
Total Data Sent	6220298	6864726	6433762	6218226	7728637	6433226	6865012	6878494	7729120
Init Elpsd Sec	205.00	264.00	251.01	204.97	301.97	250.97	263.97	264.98	301.02
Thruput Kbps	234.2	181.8	191.2	234.2	159.0	191.3	181.8	181.1	159.5
Init rttbest	1.03	0.55	0.55	1.01	0.54	0.54	0.53	0.52	0.51
Total Ack Data	31030	39464	37600	30570	44932	37484	39422	39570	44956
Resp Elpsd Sec	204.99	264.00	251.91	204.98	301.97	250.97	263.07	264.99	301.92
Resp rttbest	0.68	0.56	0.57	0.59	0.58	0.56	0.53	0.52	0.56
Dlyd Acks Rqustd	789	1031	979	789	1174	978	1029	1031	1171
Dlyd Acks Sent	791	1032	981	790	1176	979	1031	1034	1175
Est Data Pkts	6004	24002	12003	6002	48003	12002	24003	24050	48006
Chan Util Kbps	247.7	224.1	212.6	247.7	234.2	213.3	224.9	223.8	234.3
BiEf	96	87	93	96	77	93	87	87	77
Est BER	8.04E-08	3.64E-08	5.83E-08	4.02E-08	4.85E-08	3.89E-08	5.46E-08	9.12E-07	9.70E-08
IRON	7837	7837	7837	7837	7506	7506	7506	7506	7506
Date	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/5
Run No.	10	11	12	13	14	15	16	17	18
Pkt Size	125	125	250	500	125	1000	250	125	500
Start Time	17:26	17:32	17:39	17:51	18:48	18:54	18:58	19:04	19:10
Total Data Sent	7728959	7729442	6867586	6434834	7728315	6217190	6864440	7728315	6432690
Init Elpsd Sec	301.03	301.95	264.99	250.04	301.03	204.96	266.00	301.03	250.97
Thruput Kbps	159.5	159.0	181.1	192.0	159.5	234.2	180.5	159.5	191.3
Init rttbest	0.51	0.50	0.51	0.50	0.52	1.01	0.53	0.53	0.55
Total Ack Data	45110	45256	40266	37738	44756	30164	39730	44794	37374
Resp Elpsd Sec	301.93	301.96	264.99	250.94	301.04	204.97	266.00	301.04	250.08
Resp rttbest	0.52	0.50	0.51	0.50	0.52	0.57	0.53	0.53	0.54
Dlyd Acks Rqustd	1174	1172	1031	976	1173	789	1040	1174	978
Dlyd Acks Sent	1178	1179	1041	980	1173	789	1040	1174	978
Est Data Pkts	48005	48008	24012	12005	48001	6001	24001	48001	12001
Chan Util Kbps	234.2	234.2	223.4	213.4	234.9	247.7	222.4	234.9	214.1
BiEf	77	77	87	93	77	96	87	77	93
Est BER	8.09E-08	1.29E-07	2.18E-07	9.72E-08	1.62E-08	2.01E-08	1.82E-08	1.62E-08	1.94E-08

Table A-3. M-22 Experiment Data (continued)

IRON	7506	7506	7506	7506	7506	7506	7506	7506	7506
Date	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/5
Run No.	19	20	21	22	23	24	25	26	27
Pkt Size	125	1000	500	500	1000	1000	500	250	1000
Start Time	19:16	19:23	19:37	19:42	19:46	19:50	19:55	20:00	20:05
Total Data Sent	7728315	6218226	6433226	6432690	6219262	6217190	6433226	6864440	6217190
Init Elpsd Sec	301.05	204.07	250.95	250.97	204.98	204.97	250.96	265.02	204.07
Thruput Kbps	159.4	235.2	191.3	191.3	234.2	234.2	191.3	181.1	235.2
Init rttbest	0.54	0.98	0.56	0.56	1.06	1.01	0.57	0.56	1.01
Total Ack Data	44860	30576	37336	37450	30950	30192	37490	39582	30196
Resp Elpsd Sec	301.95	204.97	250.05	250.97	204.98	204.97	250.06	265.02	204.97
Resp rttbest	0.54	0.54	0.55	0.57	0.60	0.60	0.56	0.56	0.56
Dlyd Acks Rqustd	1175	790	975	980	789	789	978	1037	790
Dlyd Acks Sent	1175	791	976	980	791	789	979	1037	790
Est Data Pkts	48001	6002	12002	12001	6003	6001	12002	24001	6001
Chan Util Kbps	234.2	247.7	214.1	213.3	247.7	247.6	214.1	223.3	247.6
BiEf	77	96	93	93	96	96	93	87	96
Est BER	1.62E-08	4.02E-08	3.89E-08	1.94E-08	6.03E-08	2.01E-08	3.89E-08	1.82E-08	2.01E-08
IRON	7506	7506	7506	7506	7506	7506	7506	7837	7837
Date	12/5	12/5	12/5	12/5	12/5	12/5	12/5	12/6	12/6
Run No.	28	29	30	31	32	33	34	35	36
Pkt Size	125	500	1000	250	250	500	250	500	125
Start Time	20:10	20:23	20:27	20:32	20:37	20:42	20:47	16:46	16:51
Total Data Sent	7728798	6432690	6218226	6864440	6864726	6432690	6865012	6442338	7730730
Init Elpsd Sec	301.98	250.99	204.07	263.99	266.02	251.02	264.02	249.98	301.91
Thruput Kbps	159.0	191.2	235.2	181.8	180.4	191.2	181.8	192.0	159.0
Init rttbest	0.56	0.58	1.03	0.56	0.57	0.58	0.57	0.55	0.53
Total Ack Data	45000	37422	30538	39426	39802	37526	39608	38900	45522
Resp Elpsd Sec	301.98	250.99	204.97	264.00	266.02	251.02	264.92	249.99	301.02
Resp rttbest	0.61	0.56	0.56	0.57	0.56	0.61	0.58	0.55	0.58
Dlyd Acks Rqustd	1173	980	789	1032	1040	982	1032	970	1164
Dlyd Acks Sent	1176	980	790	1032	1041	982	1034	987	1179
Est Data Pkts	48004	12001	6002	24001	24002	12001	24003	12019	48016
Chan Util Kbps	234.2	213.3	247.7	224.1	222.4	213.3	223.4	214.5	235.0
BiEf	77	93	96	87	87	93	87	93	77
Est BER	6.47E-08	1.94E-08	4.02E-08	1.82E-08	3.64E-08	1.94E-08	5.46E-08	3.69E-07	2.59E-07

Table A-3. M-22 Experiment Data (continued)

IRON	7837	7837	7837	7837	7837	7837	7837	7837	7506
Date	12/6	12/6	12/6	12/6	12/6	12/6	12/6	12/6	12/6
Run No.	37	38	39	40	1	2	3	4	S1
Pkt Size	1000	125	1000	250	1000	250	500	1000	1000
Start Time	16:57	17:01	17:06	17:11	17:33	17:38	17:44	17:49	19:10
Total Data Sent	6230658	7732823	6263848	6875594	6326008	6898760	6489506	6362268	6218226
Init Elpsd Sec	205.01	301.00	207.01	263.99	209.01	265.02	247.94	214.94	204.08
Thruput Kbps	234.1	159.5	231.9	181.8	229.7	181.1	193.6	223.3	235.2
Init rttbest	1.03	0.52	0.52	0.52	0.53	0.50	0.51	0.51	0.88
Total Ack Data	34788	46056	45396	41538	62930	52400	44870	71776	16626
Resp Elpsd Sec	205.01	301.00	207.92	263.99	209.01	265.02	247.05	214.04	204.98
Resp rttbest	0.53	0.53	0.56	0.52	0.50	0.51	0.50	0.49	0.59
Dlyd Acks Rqustd	783	1161	778	1006	754	969	913	754	430
Dlyd Acks Sent	796	1185	823	1044	857	1087	1015	897	431
Est Data Pkts	6014	48029	6046	24040	6106	24121	12107	6141	6002
Chan Util Kbps	248.1	235.1	246.0	224.5	247.1	224.4	218.6	242.7	247.7
BiEf	96	77	95	87	94	86	92	93	96
Est BER	2.81E-07	4.69E-07	9.22E-07	7.28E-07	2.11E-06	2.20E-06	2.07E-06	2.80E-06	4.02E-08
IRON	7506	7506	7506	7506	7506	7506	7506	7506	7506
Date	12/6	12/6	12/6	12/6	12/6	12/6	12/6	12/6	12/6
Run No.	S2	S3	S4	5	6	7	8	9	10
Pkt Size	1000	1000	1000	125	500	250	250	125	125
Start Time	19:15	19:38	19:43	19:49	19:55	20:00	20:06	20:19	20:26
Total Data Sent	6218226	6276280	6265882	7737814	6446090	6876738	6881600	7747635	7744898
Init Elpsd Sec	204.97	209.97	206.04	300.95	249.03	261.97	259.99	300.00	301.02
Thruput Kbps	234.2	228.6	233.0	159.5	192.8	183.2	184.6	160.0	159.5
Init rttbest	0.89	0.58	0.83	0.55	0.56	0.56	0.56	0.56	0.56
Total Ack Data	16912	32736	28174	47436	39346	41980	42896	50532	56202
Resp Elpsd Sec	204.97	209.08	206.94	300.96	249.93	261.97	259.99	300.00	301.92
Resp rttbest	0.59	0.55	0.60	0.56	0.58	0.56	0.58	0.61	0.61
Dlyd Acks Rqustd	433	430	415	1143	959	999	984	1114	1123
Dlyd Acks Sent	434	487	460	1195	984	1041	1042	1223	1218
Est Data Pkts	6002	6058	6048	48060	12026	24044	24061	48121	48104
Chan Util Kbps	247.7	245.1	247.2	235.3	214.7	226.3	228.2	236.3	234.7
BiEf	96	95	95	77	93	87	87	77	77
Est BER	4.02E-08	1.16E-06	9.61E-07	9.70E-07	5.05E-07	8.01E-07	1.11E-06	1.95E-06	1.68E-06

Table A-3. M-22 Experiment Data (continued)

IRON	7506	7506	7506	7506	7506	7506	7506	7506	7506
Date	12/6	12/6	12/6	12/6	12/6	12/6	12/6	12/6	12/6
Run No.	11	12	13	14	15	16	17	18	19
Pkt Size	125	250	500	125	1000	250	125	500	125
Start Time	20:31	20:38	20:42	20:48	20:54	20:59	21:12	21:18	21:23
Total Data Sent	7747796	6894470	6492186	7763735	6333222	6896186	7753914	6478786	7757494
Init Elpsd Sec	300.99	261.01	246.96	299.01	210.05	260.96	299.05	248.95	300.00
Thruput Kbps	159.5	183.9	194.4	160.5	228.5	183.9	160.5	192.8	160.0
Init rttbest	0.56	0.56	0.57	0.56	1.02	0.56	0.56	0.57	0.55
Total Ack Data	56862	45788	45348	60054	66148	46062	53054	46868	53044
Resp Elpsd Sec	300.99	261.01	246.06	299.01	210.06	260.97	299.95	248.05	300.00
Resp rttbest	0.61	0.58	0.56	0.61	0.56	0.58	0.60	0.55	0.55
Dlyd Acks Rqustd	1116	965	905	1061	749	959	1096	932	1082
Dlyd Acks Sent	1224	1067	1013	1263	863	1066	1248	1014	1242
Est Data Pkts	48122	24106	12112	48221	6113	24112	48160	12087	48182
Chan Util Kbps	235.5	227.7	219.6	237.6	246.2	227.8	236.6	217.4	236.6
BiEf	77	86	92	77	94	86	77	92	77
Est BER	1.97E-06	1.93E-06	2.17E-06	3.57E-06	2.25E-06	2.03E-06	2.58E-06	1.68E-06	2.94E-06
IRON	?	?	?	?	?	?	7837	7837	7837
Date	12/7	12/7	12/7	12/7	12/7	12/7	12/7	12/7	12/8
Run No.	20	21	22	23	24	25	26	27	28
Pkt Size	1000	500	500	1000	1000	500	250	1000	125
Start Time	12:16	12:24	12:30	12:35	12:41	12:46	23:44	23:49	17:45
Total Data Sent	6404744	6627794	6616538	6550782	6593296	6737712	6873306	6238946	7843468
Init Elpsd Sec	218.99	249.98	244.00	224.97	227.96	252.03	262.99	205.03	303.00
Thruput Kbps	219.2	192.0	196.7	213.4	210.6	190.5	182.5	234.1	158.4
Init rttbest	0.54	0.51	0.51	0.57	0.52	0.48	0.47	1.03	0.48
Total Ack Data	80002	81440	62982	89760	106804	109524	41864	37374	123066
Resp Elpsd Sec	218.08	249.99	244.01	224.98	227.96	252.03	263.00	205.03	303.01
Resp rttbest	0.51	0.53	0.49	0.55	0.49	0.48	0.47	0.47	0.48
Dlyd Acks Rqustd	747	665	678	387	400	634	1035	782	875
Dlyd Acks Sent	928	1008	996	685	741	1161	1064	803	1513
Est Data Pkts	6182	12365	12344	6323	6364	12570	24032	6022	48716
Chan Util Kbps	239.8	220.6	225.7	237.7	236.1	222.5	225.3	248.4	236.9
BiEf	93	89	90	90	90	88	87	96	75
Est BER	3.61E-06	6.99E-06	6.59E-06	6.33E-06	7.11E-06	1.08E-05	5.82E-07	4.42E-07	1.15E-05

Table A-3. M-22 Experiment Data (continued)

IRON	7837	7837	7837	7837	7837	7837	7837	7837	7837
Date	12/8	12/12	12/12	12/12	12/12	12/13	12/13	12/13	12/13
Run No.	29	5	2	3	1	4	6	7	8
Pkt Size	500	125	250	500	1000	1000	500	250	250
Start Time	17:51	17:10	17:27	17:35	17:52	16:29	16:38	16:44	16:52
Total Data Sent	6779482	7737331	6887320	6482538	7537092	6323898	6497584	6929114	6933652
Init Elpsd Sec	253.99	301.00	261.04	247.94	339.98	212.00	247.99	263.04	265.07
Thruput Kbps	189.0	159.5	183.9	193.6	141.2	226.4	193.6	182.5	181.1
Init rttbest	0.49	0.49	0.48	0.49	0.45	1.20	0.53	0.52	0.51
Total Ack Data	137440	47666	44080	46876	191062	65842	48956	63812	71154
Resp Elpsd Sec	253.99	300.99	261.93	247.04	339.08	212.00	248.00	263.04	265.97
Resp rttbest	0.49	0.54	0.51	0.47	0.44	0.57	0.51	0.52	0.53
Dlyd Acks Rqustd	695	1145	983	921	862	758	908	921	925
Dlyd Acks Sent	1277	1199	1062	1013	1804	864	1025	1135	1155
Est Data Pkts	12648	48057	24081	12094	7275	6104	12122	24227	24243
Chan Util Kbps	222.1	235.2	226.7	218.4	181.5	243.5	218.1	227.1	224.7
BiEf	87	77	87	92	78	94	92	86	86
Est BER	1.23E-05	9.21E-07	1.47E-06	1.82E-06	2.32E-05	2.07E-06	2.36E-06	4.12E-06	4.40E-06
IRON	7837	7837	7837	7837	7506	7506	7506	7506	7506
Date	12/13	12/13	12/13	12/13	12/13	12/13	12/13	12/13	12/13
Run No.	9	10	11	12	13	14	15	16	17
Pkt Size	125	125	125	250	500	125	1000	250	125
Start Time	16:59	17:07	17:32	17:40	20:20	20:26	20:33	20:40	20:47
Total Data Sent	7795973	7796579	7781483	6996038	6686256	7850552	6739372	7054668	7839604
Init Elpsd Sec	303.00	311.00	301.08	266.00	255.02	313.91	249.04	274.04	310.99
Thruput Kbps	158.4	154.3	159.4	180.4	188.2	152.9	192.7	175.2	154.3
Init rttbest	0.50	0.49	0.47	0.46	0.57	0.56	0.56	0.56	0.55
Total Ack Data	85342	126576	77286	93638	113578	151538	134886	151702	149260
Resp Elpsd Sec	303.00	311.00	301.98	266.01	255.02	313.02	249.95	274.94	310.99
Resp rttbest	0.50	0.49	0.46	0.46	0.57	0.56	0.59	0.56	0.55
Dlyd Acks Rqustd	978	1032	1030	833	759	920	710	774	923
Dlyd Acks Sent	1364	1405	1341	1261	1194	1594	1160	1383	1560
Est Data Pkts	48421	48425	48331	24461	12474	48760	6505	24666	48692
Chan Util Kbps	235.4	229.4	235.8	226.7	218.2	229.5	220.1	221.2	230.7
BiEf	76	76	76	85	88	75	87	83	75
Est BER	6.78E-06	6.84E-06	5.34E-06	8.32E-06	9.04E-06	1.22E-05	9.75E-06	1.20E-05	1.11E-05

Table A-3. M-22 Experiment Data (continued)

IRON	7837	7837	7837	7837	7837	7837	7837	7837	7837
Date	12/14	12/14	12/15	12/15	12/15	12/15	12/15	12/15	12/15
Run No.	18	19	20	21	22	23	24	25	26
Pkt Size	500	125	1000	500	500	1000	1000	500	250
Start Time	16:24	17:14	16:30	16:37	16:45	16:53	17:11	17:16	17:23
Total Data Sent	6820256	7860856	6219262	6433226	6433226	6224442	6224442	6435370	6867300
Init Elpsd Sec	266.05	320.98	205.02	250.96	250.91	205.06	204.98	250.96	265.05
Thruput Kbps	180.4	149.5	234.1	181.3	191.3	234.1	234.2	191.3	181.1
Init rttbest	0.52	0.48	1.08	0.52	0.52	1.01	1.02	0.49	0.47
Total Ack Data	153780	199784	31104	37490	37408	31568	32296	37842	40250
Resp Elpsd Sec	266.96	320.09	205.02	251.00	250.01	205.96	204.99	250.06	265.95
Resp rttbest	0.52	0.48	0.57	0.55	0.51	0.55	0.54	0.48	0.48
Dlyd Acks Rqustd	702	913	791	978	976	792	787	976	1035
Dlyd Acks Sent	1362	1675	793	979	977	797	793	981	1045
Est Data Pkts	12724	48824	6003	12002	12002	6008	6008	12006	24011
Chan Util Kbps	212.6	224.7	247.7	213.3	214.2	246.7	247.9	214.2	222.6
BiEf	86	74	96	93	93	96	96	93	87
Est BER	1.37E-05	1.32E-05	6.03E-08	3.89E-08	3.89E-08	1.61E-07	1.61E-07	1.17E-07	2.00E-07

Table A-3. M-22 Experiment Data (concluded)

IRON	7837	7837	7837	7837	7837
Date	12/15	12/15	12/15	12/15	12/15
Run No.	27	28	29	30	31
Pkt Size	1000	125	500	1000	250
Start Time	17:28	17:32	17:38	17:43	17:48
Total Data Sent	6225478	7736043	6473962	6421282	7229128
Init Elpsd Sec	204.99	300.96	248.02	213.03	314.02
Thruput Kbps	234.2	159.5	193.5	225.3	152.9
Init rttbest	0.96	0.46	0.46	0.48	0.44
Total Ack Data	32842	47974	43332	89690	291790
Resp Elpsd Sec	204.09	300.96	248.02	213.93	314.93
Resp rttbest	0.46	0.46	0.47	0.49	0.44
Dlyd Acks Rqustd	789	1172	954	715	757
Dlyd Acks Sent	796	1215	1028	911	1895
Est Data Pkts	6009	48049	12078	6198	25276
Chan Util Kbps	249.1	235.2	217.2	245.1	197.9
BiEf	96	77	92	92	80
Est BER	1.81E-07	7.92E-07	1.51E-06	3.92E-06	2.26E-05

A.2 LINK PERFORMANCE DATA

The link performance data for the 25b Kbps downlink was recorded at periodic intervals during each contact. A set of measurements were recorded prior to the start of the data runs, periodically throughout the contact, and a final set at the end of the contact. Table A-1 summarizes the data for all contacts during which valid experiment data was obtained. The columns have the following meanings:

- Date: The day in December 1995 when the measurement was made.
- Time: The Greenwich Mean Time (GMT) of the measurement.
- Err/5 Min: The number of errors detected by the BER test set in 5 minutes. Test data rate was 256 Kbps.
- BER: The error rate calculated from the BERT. See Equation B.3-1.
- Az: Antenna azimuth at time of measurement.
- El: Antenna elevation at time of measurement.
- Range NM: Slant range in nautical miles from ground station to spacecraft at time of measurement.

- 2 Way Delay (sec): The round trip propagation time, in seconds. See Equation B.3-5.
- Sig Str (Neg dBm): The signal strength of the link, measured in dBm. The values are negative, so a value of 103 is a greater signal level than 104.
- Mod Index Adjust: Indicates whether or not the modulation index was adjusted to a value other than normal prior to the measurement.
- IRON: The spacecraft being tracked.

Table A-4. Summary of Link Performance Data

Date	Time	Err/5 Min	BER*	Az	El	Range NM	2 Way Delay (sec)	Sig Str (Neg dBm)	Mod Index Adjust	IRON
12/5	16:05	0	1.00E-08	316.8	35.5	22639	0.279516	106	No	7837
12/5	17:05	3	3.90E-08	311.8	30.6	21436	0.264663	107	No	7837
12/5	17:44	8	1.00E-07	308.4	26.4	20130	0.248538	108	No	7837
12/5	18:15	0	1.00E-08	306.8	38.2	19906	0.245773	103	No	7506
12/5	19:27	9	1.17E-07	315.8	34.8	22077	0.272577	104	No	7506
12/5	20:15	13	1.69E-07	320.4	33.4	22759	0.280998	105	No	7506
12/5	20:52	0	1.00E-08	323.1	32.8	22816	0.281702	105	No	7506
12/6	16:40	60	7.81E-07	313.4	32.5	21933	0.270799	106	No	7837
12/6	17:20	27	3.52E-07	310.1	28.6	20853	0.257465	108	No	7837
12/6	17:27	141	1.84E-06	309.2	27.5	20514	0.253280	108	Yes	7837
12/6	17:54	355	4.62E-06	306.5	23.8	19318	0.238513	109	Yes	7837
12/6	19:05	3	3.91E-08	310.5	36.3	20884	0.257848	103	No	7506
12/6	19:25	105	1.37E-06	316.4	34.7	22190	0.273973	104	Yes	7506
12/6	20:11	225	2.93E-06	320.5	33.4	22768	0.281109	105	Yes	7506
12/6	21:04	260	3.39E-06	323.9	32.6	22742	0.280788	104	Yes	7506
12/6	21:30	262	3.41E-06	325.3	32.5	22407	0.276652	104	Yes	7506
12/7	23:35	195	2.54E-06	30.2	11.9	17917	0.221215	105	Yes	7837
12/7	23:56	41	5.34E-07	28.6	12.6	19106	0.235895	105	Yes	7837
12/8	17:20	437	5.69E-06	NM	NM	NM	NM	NM	No	7837
12/8	18:25	97574	1.27E-03	NM	NM	NM	NM	NM	No	7837
12/12	16:57	45	5.86E-07	309.6	27.8	20700	0.255576	106	NM	7837
12/12	17:45	338	4.40E-06	305.3	22	18768	0.231722	109	NM	7837
12/12	18:08	70258	9.15E-04	302.5	18	17454	0.215499	NM	NM	7837
12/13	16:20	896	1.17E-05	311.6	30.4	21409	0.264330	105	Yes	7837
12/13	17:15	2346	3.05E-05	307.6	25.3	19833	0.244871	107	Yes	7837
12/13	17:25	1220	1.59E-05	306.5	23.7	19336	0.238735	107	Yes	7837
12/13	17:55	14575	1.90E-04	303.3	19.2	17836	0.220215	111	Yes	7837

12/13	20:05	765	9.96E-06	322.1	32.9	22848	0.282097	103	Yes	7506

Table A-4. Summary of Link Performance Data (concluded)

12/13	20:53	697	9.08E-06	325.0	32.4	22510	0.277923	103	Yes	7506
12/14	16:10	898	1.17E-05	312.9	31.9	21799	0.269145	105	Yes	7837
12/14	16:45	1091	1.42E-05	310.0	28.5	20822	0.257082	106	Yes	7837
12/14	17:05	899	1.17E-05	302.8	26.1	20109	0.248279	106	Yes	7837
12/14	17:50	NM	NM	303.8	19.8	18055	0.222919	109	Yes	7837
12/14	18:00	NM	NM	302.6	18.1	17507	0.216153	111	No	7837
12/14	18:08	NM	NM	301.5	16.6	17004	0.209943	113	No	7837
12/14	18:15	826175	1.08E-02	NM	NM	NM	NM	114	No	7837
12/15	16:05	3	3.91E-08	312.8	31.9	21809	0.269268	105	No	7837
12/15	17:00	19	2.47E-07	307.7	25.5	19919	0.245933	107	No	7837
12/15	17:30	NM	NM	NM	NM	18833	0.232525	108	No	7837
12/15	17:35	NM	NM	NM	NM	18584	0.229450	109	No	7837
12/15	17:50	NM	NM	NM	NM	17809	0.219882	110	No	7837
12/15	17:57	6639	8.64E-05	302.4	17.8	17418	0.215054	111	No	7837

* When 0 errors were measured, a BER value of 1.00E-08 was assigned.
Note: NM=Not Measured or Not Calculated

We used the BER measurements to help establish confidence in the error estimates derived from the number of retransmitted packets described in Section 6. Establishment of this confidence then allowed the error estimate based on packet retransmission count to be used as the estimator of error during each data run. We consider this procedure for determining error rates more consistent than using the values obtained by interpolating between the BER measurements. As shown in Table A-4, the variation in BER for any given signal level could vary by as much as a significant fraction of an order of magnitude, especially at the lower error rates. The reason for this is that at the lower error rates and the relatively short measurement period, the probability of counting an error was very small. When an error were counted at these relatively high signal levels, it constituted a relatively significant change in the BER. For example, the measurement at 16:40 on 6 December gave a BER of 7.8×10^{-7} for a signal level of -106 dBm, while other measurements at this signal level gave zero errors. Just 40 minutes later, the signal level had decreased to -107.5 dBm, yet the error count was half that of earlier measurement. In terms of percent of errored bits, the difference between the two measurements is only 0.000043 % of the total number of bits transmitted in five minutes. This was the widest dispersion noted in the measurement data. Figure A-1 shows the plot of the BER measurements against signal level made without adjustments to the modulation index. The regression line shows the fit for the data from IRON 7837; there was not sufficient signal strength range for the 7506 data to compute a reliable regression line. The coefficient of regression for the exponential line through the 7837 data is 0.93.

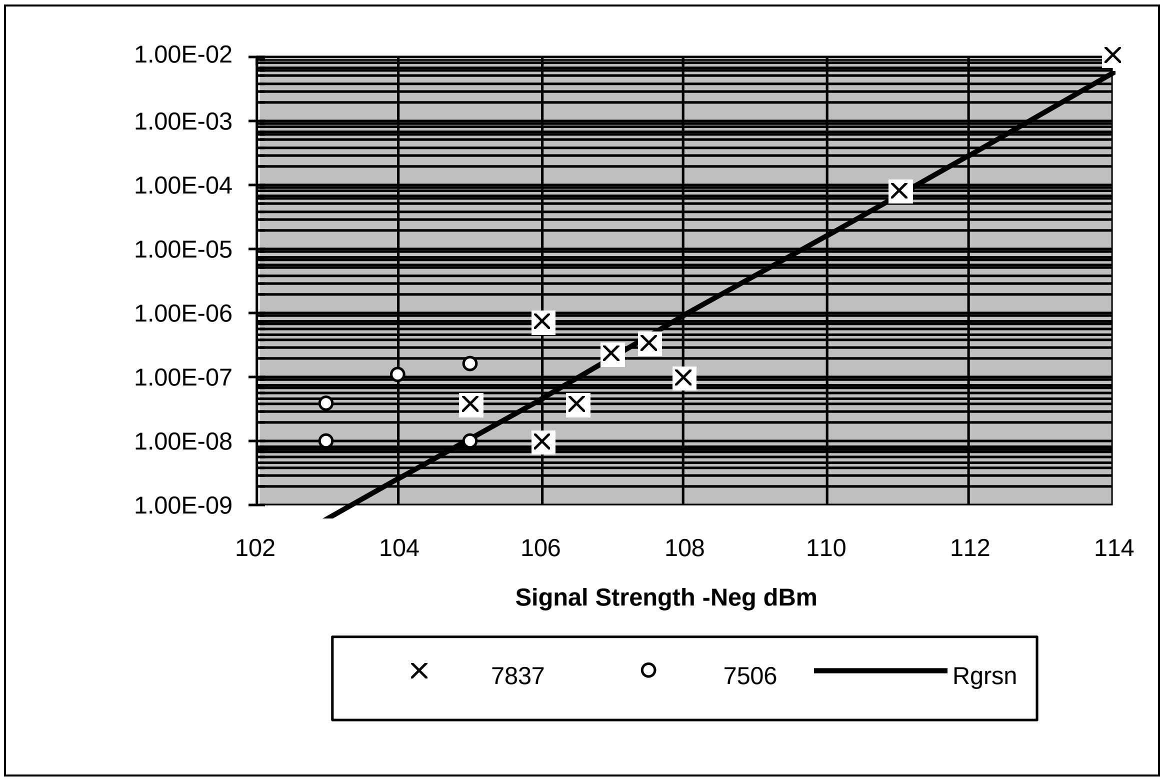
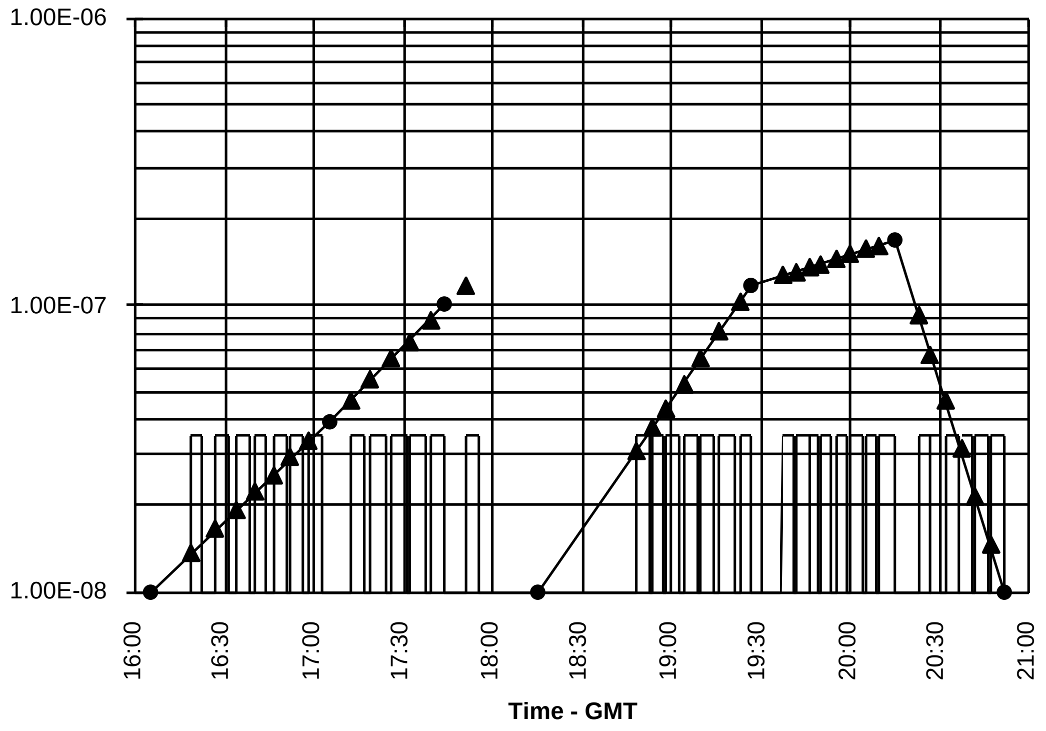
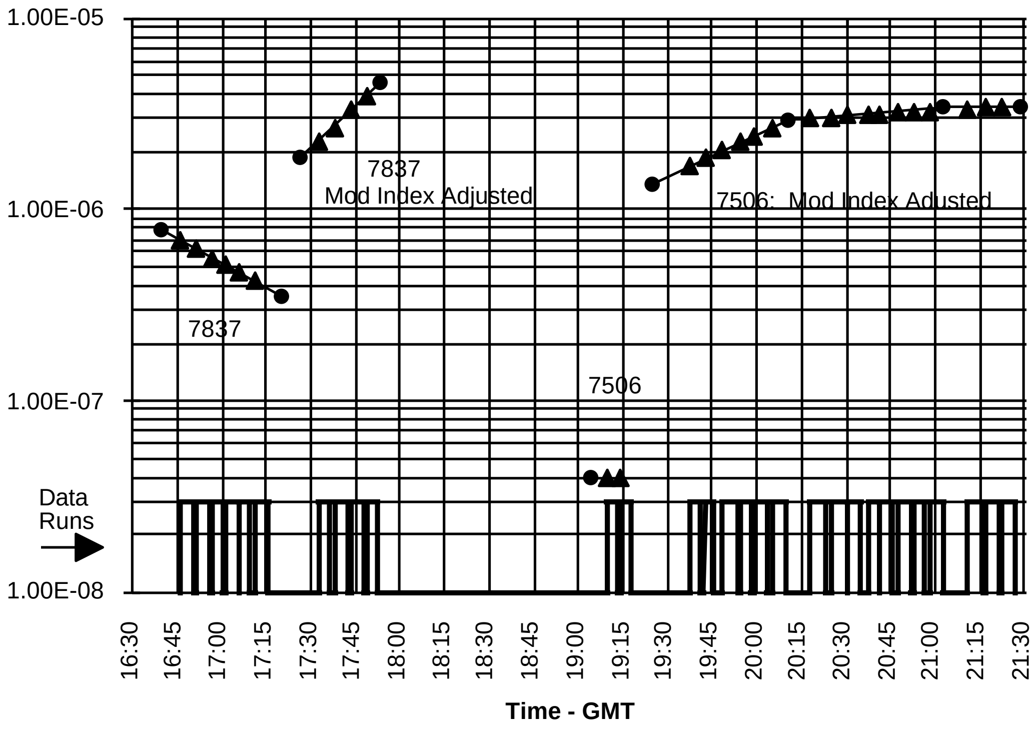
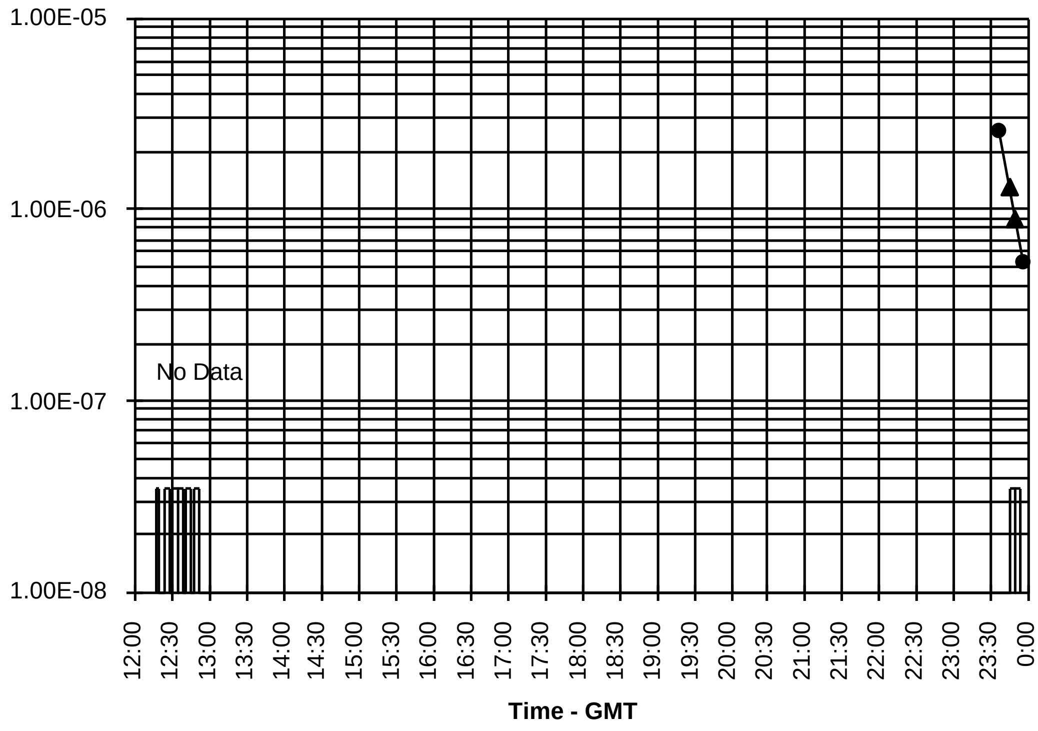
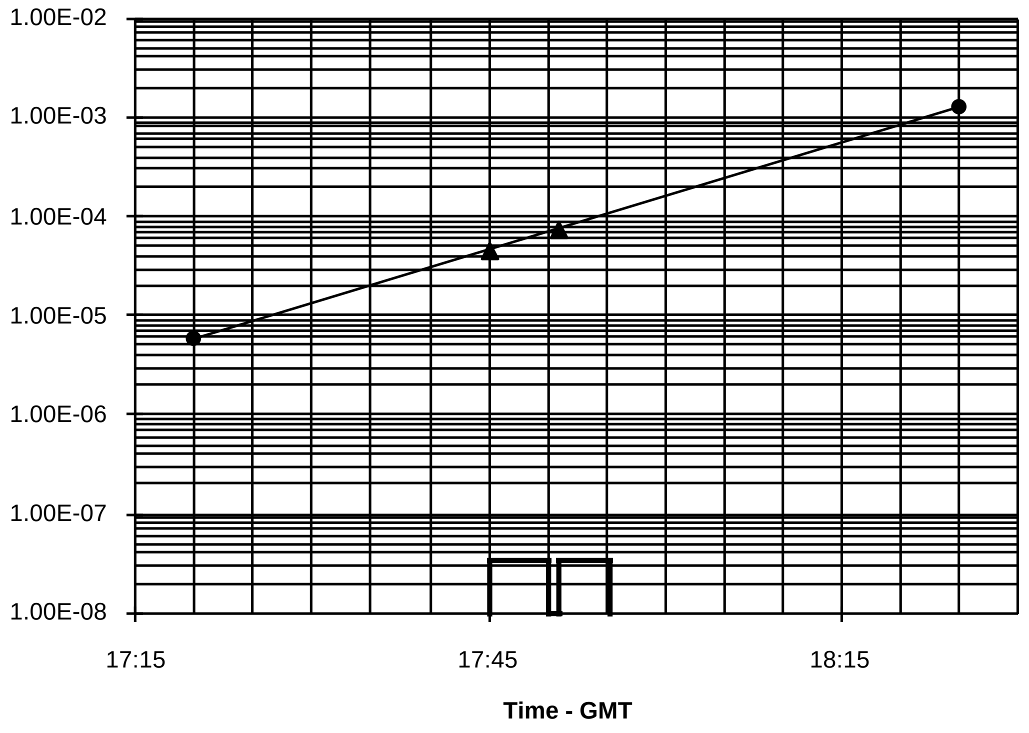


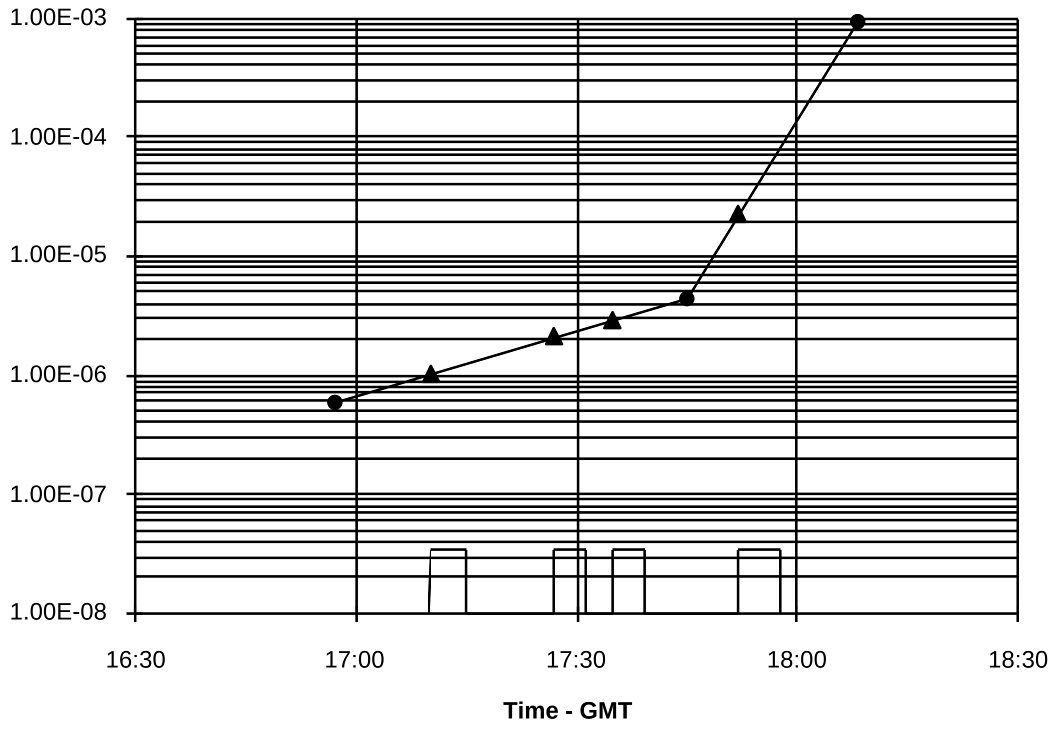
Figure A-1. Bit Error Rate versus Signal Strength

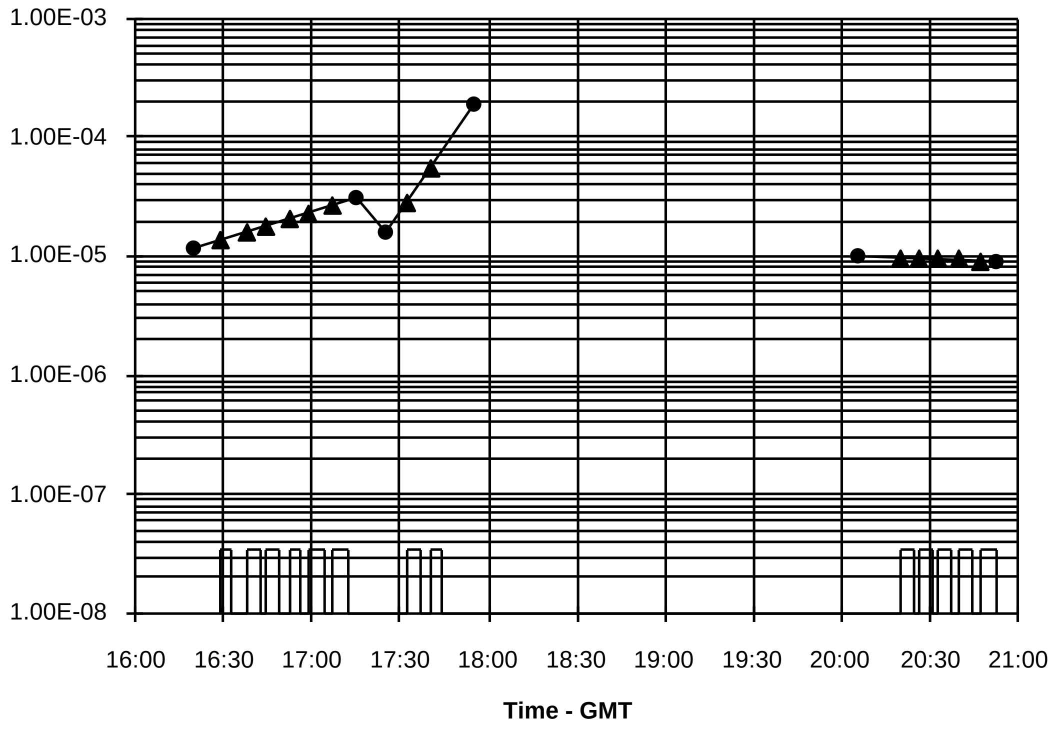

1.00E-06
1.00E-07
1.00E-08
16:00
16:30
17:00
17:30
18:00
18:30
19:00
19:30
20:00
20:30
21:00
Time - GMT

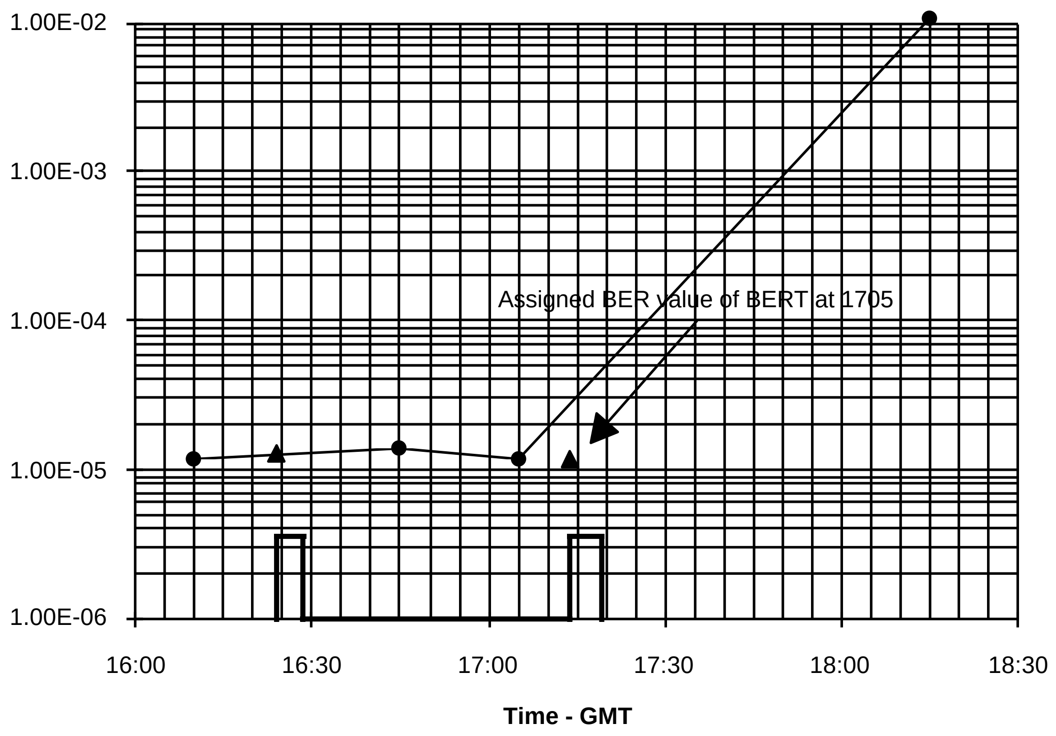

1.00E-05
1.00E-06
1.00E-07
1.00E-08
Data Runs
7837
7837 Mod Index Adjusted
7506
7506: Mod Index Adusted
16:30
16:45
17:00
17:15
17:30
17:45
18:00
18:15
18:30
18:45
19:00
19:15
19:30
19:45
20:00
20:15
20:30
20:45
21:00
21:15
21:30
Time - GMT

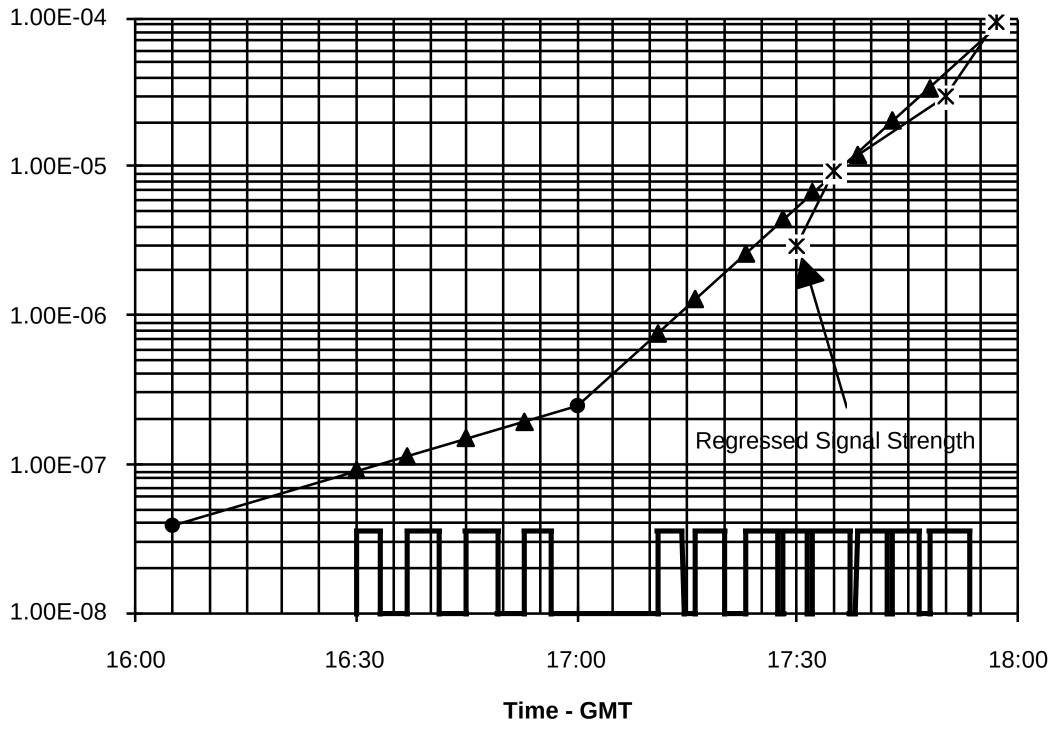

1.00E-05
1.00E-06
1.00E-07
1.00E-08
No Data
12:00
12:30
13:00
13:30
14:00
14:30
15:00
15:30
16:00
16:30
17:00
17:30
18:00
18:30
19:00
19:30
20:00
20:30
21:00
21:30
22:00
22:30
23:00
23:30
0:00
Time - GMT


1.00E-02
1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
17:15
17:45
18:15
Time - GMT


1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
16:30
17:00
17:30
18:00
18:30
Time - GMT


1.00E-03
1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
16:00
16:30
17:00
17:30
18:00
18:30
19:00
19:30
20:00
20:30
21:00
Time - GMT


1.00E-02
1.00E-03
1.00E-04
1.00E-05
1.00E-06
16:00
16:30
17:00
17:30
18:00
18:30
Time - GMT
Assigned BER value of BERT at 1705


1.00E-04
1.00E-05
1.00E-06
1.00E-07
1.00E-08
Regressed Signal Strength
16:00
16:30
17:00
17:30
18:00
Time - GMT

APPENDIX B: EQUATIONS

B.1 General Equations $\boxed{2^k r}$

Equation B.1-1: Confidence Interval (I_α)

$$I_\alpha = \mu \pm z_{1-\alpha/2} (s/\sqrt{n}) \qquad \text{Eq B.1-1}$$

where α = 1 - (desired confidence interval)

μ = mean of the data

$z_{1-\alpha/2}$ = Value of standard Normal coordinate corresponding to a cumulative probability of 1 - $\alpha/2$

s = sample standard deviation

n = sample size

The following example illustrates the use of the equation:

Assume a confidence level of 90% is desired for a set of data with mean 233, standard deviation of 6, and sample size of 9.

Then $\alpha = 1 - 0.90$

$= 0.10$

$\mu = 233$

$z_{1-\alpha/2}$ = 1.645 (from any table of standard Normal values)

$s = 6$

$n = 9$

and $I_\alpha = 233 \pm 1.645(6/\sqrt{9})$

$= 233 \pm 3.29$

$= (229.71, 236.29)$

B.2 Equations for Analysis of Protocol Performance

Equation B.2-1: Throughput

$$\text{Throughput}=(\text{user data transmitted})/(\text{time spent in data transfer phase}) \qquad \text{Eq B.2-1}$$

Equation B.2-2: Channel Utilization

Channel Utilization (Kbps) = (total data sent)/(connection duration) •
(1/Channel Data Rate in Kbps) Eq B.2-2

Equation B.2-3: Bit Efficiency

Bit Eff.=(user data transmitted)/(total initiator data + total responder data)
Eq B.2-3

Equation B.2-4: Estimated Data Packets (Pkt_{Tot})

$$Pkt_{Tot} = (N\text{-}Oh)/(\,P_i+Dh) \qquad \text{Eq B.2-4}$$

where N = Total amount of data sent, in bytes
Oh = size of connect/disconnect packets, in bytes
P_i = size of data packet type i, in bytes
(P_1 = 125, P_2 = 250, P_3 = 500, P_4 = 1000)
Dh = size of data packet overhead, in bytes
= 20 + 16 for TCP compressed header with Timestamp
= 20 + 32 for TCP uncompressed header with Timestamp
(20 = size of IP header)

and Oh = SYN + Handshake ACK + FIN
where SYN = Connection request packet
= 20 + 48
(48 = size of TCP header with options)
(20 = size of IP header)
Handshake ACK = 20 + 32
32 = size of TCP header with Timestamp
(20 = size of IP header)
FIN = Connection close request
= 20 + 14 for TCP compressed header with Timestamp
= 20 + 32 for TCP uncompressed header with Timestamp
(20 = size of IP header)

The following example illustrates the use of the equation:

Let N= 7728959 bytes of total data sent using compressed headers and data packet type 1

Then Oh = (20+48) + (20+32) + (20+14)
= 154
and P_1 = 125
and Dh = 20 + 16
= 36
So Pkt_{Tot} = (7728959-154)/(125+36)
= 40893.15
≅ 40893 packets sent

B.3 Equations for Analysis of Link Performance and Delay

Note: Logarithms to base 10 are denoted by *log*. Logarithms to base *e* are denoted by *ln*.

Equation B.3-1: Link Bit Error Rate (BER)

BER=(errors in 5 min)/(256,000*60*5) Eq B.3-1

Equation B.3-2: Link Signal Strength Change as a Function of Change in Range (SS_{Rng})

$$SS_{Rng} = 20\log(r2/r1) \qquad \text{Eq B.3-2}$$

where r1 = initial range
r2 = final range

Equation B.3-3: Link Bit Error Rate Predicted by Log-Linear Interpolation (BER)

The estimate of bit error rate for each run was initially made by linear interpolation between two bit error rate measurements plotted on a logarithmic scale. Since the bit error rates are stated as powers of ten, the line between two measurements is actually a logarithmic curve if plotted on a linear scale. Therefore, the form of the equation is an exponential, as follows:

$$BER = bm^x \qquad \text{Eq B.3-3}$$

Where x= time (stated in fractions of a day)
b and m are constants

To estimate the parameters b and m:

Let $y' = \log y = \log(bm^{x})$
$= \log b + x\log m$
$= m'x+b'$

Let y_1= Value of BERT at time x_1
and y_2= Value of BERT at time x_2
Then $y'_1 = \log (y_1)$
and $y'_2 = \log (y_2)$
Then $m' = (y'_2 - y'_1)/ (x_2 - x_1)$
and $b' = y'_1 - mx_1$

The following example illustrates the use of the equation:

Let $y_1 = 8 \times 10^{-7}$: $x_1 = 0$
Let $y_2 = 3.5 \times 10^{-7}$: $x_2 = 1/24$ (i.e., 1 hour)
Then $y_1' = \log(y_1) = -6.097$
$y_2' = \log(y_2) = -6.456$
So $m' = (-6.456+6.097)/(1/24)$
$= -0.359 (24)$
$= -8.617$
and $b' = y_1' = -6.097$ (note that $x_1 = 0$)

So $y' = -8.617x-6.097$

To use this equation of the line between y_1' and y_2' to estimate a BER value for a run between times x_1 and x_2, substitute the time (as a fraction of a day) into the equation, solve, and take the antilogarithm. For example:

Let $x = 1/48$ (i.e., 30 minutes after x_1)
Then $y' = -8.616(1/48)-6.097$
$= -6.276$
And $y = \text{alog}(-6.276)$
$= 5.3 \times 10^{-7}$

Equation B.3-4: Link Round Trip Propagation Delay

The SCPS-TP protocol is designed to work reliably over the long delays encountered in space communications. As a consequence, it was helpful during data analysis to know the round trip propagation delay. The slant range from the antenna to the satellite was recorded when error rate measurements were made. The range measurement was given in units of nautical miles. The round trip delay is given by the following equation:

$$\text{Delay} = 3704r/c \qquad \text{Eq B.3-4}$$

Where 3704 = 2 (1852 meters/nautical mile)
r= range in nautical miles
c= speed of light
= 300,000,000 meters/second

So, for example, a range of 20,000 nautical miles yields a round trip light delay of**:**

3704 x 20,000 / 300,000,000 = 0.247 seconds

APPENDIX C: SCPS SUPPORT REPORT

The following, “SCPS Support Report,” is reproduced in its entirety.

APPENDIX D: SCPS PROTOCOL OVERVIEW

The SCPS Transport protocol is based on TCP. In fact, SCPS-TP is essentially standard TCP augmented by a set of extensions and enhancements that consist of both implementation and specification changes. These modifications each respond to requirements derived from characteristics of the space environment, the mission communication scenarios, and other driving factors. Some of the constraints imposed by the space environment that led to TCP modifications include:

- Space link delays ranging from milliseconds to hours.
- Potentially noisy space links.
- Limited space link bandwidth.
- Limited periods of connectivity.
- A mismatch between the up-link and down-link channel capacities.
- Limited onboard processing power and memory (for programs and data buffering).
- Link interruptions caused by bursts of noise and antenna obscurations.

A number of TCP extensions have been defined to address these and other requirements. Some of these modifications were proposed by members of the research community and were adopted by the SCPS developers, while other enhancements were designed by the SCPS team, in some cases by drawing from the work of others. The following subsections describe some of the major SCPS extensions to TCP. The discussion of each feature contains a brief motivation or description of the constraint being addressed, a comparison to other similar enhancements that have been proposed if applicable, and a synopsis of the SCPS extension itself. The interested reader can refer to the SCPS-TP specification [5] for details on the SCPS extensions beyond those presented here.

D.1 IDENTIFYING THE SOURCE OF PACKET LOSS

RFC 1106 [10] raises the important issue of differentiating between packet loss due to congestion and loss due to corruption. Because the appropriate response to congestion is quite different from the proper response to corruption, distinction between the two is essential when operating in an environment where both events are possible. However, the problem of identifying the source of packet loss can be generalized beyond simply differentiating between congestion and corruption. Different communications environments may be subject to loss caused by a variety of factors for which the appropriate responses differ, such as link outage or mobility, as well as network congestion and noise. A brief list of possible sources of packet loss includes:

- intermittent connectivity due to the ability to communicate with a satellite only during the portion of its orbit when it is in view.
- atmospheric interference that results in corruption.
- network congestion.
- losses due to hand-offs in a cellular communication network.
- antenna obscurations due to terrain features in line-of-site wireless communication.

The appropriate responses to these sources of packet loss differ. In the case of congestion, the correct response is to reduce the transmission rate. In the case of corruption, the appropriate response is to continue transmitting in the hope that some packets will reach the destination. In the case of a temporary link outage, the best response is to suspend transmission and wait for the link to be restored. As mentioned earlier, TCP fails to make the distinction between congestion and any other source of loss. Instead, TCP assumes that all packet loss is a result of network congestion, and reacts by invoking its congestion control algorithms. These algorithms immediately reduce the transmission rate drastically, and then gradually increase the rate again as long as no further loss occurs. This action allows the pipe to drain somewhat so that the congestion can subside. This approach is effective at controlling congestion, and at worst, results in oscillating behavior between congestion periods and nearly idle periods. However, this mechanism, which is triggered by any packet loss, reduces throughput and provides absolutely no benefit when the loss experienced is a result of noise, link outage, or changing connectivity.

The SCPS extensions provide a mechanism to change TCP's default assumption as to the source of segment loss from congestion to corruption. TCP relies on this default assumption in the absence of any other information about the state of the network. A network manager has the ability to set the default packet loss assumption appropriately for a particular network based on the most likely event. In addition, to decrease reliance on the default response, three signals are defined that provide explicit notification to TCP about the source of loss. These signals are the link outage signal, which indicates that the link is temporarily unavailable; the source quench, which signals the presence of congestion; and the corruption experienced signal, which notifies TCP of loss due to noise. Each of these signals evokes the appropriate response from TCP regardless of the default setting.

The congestion versus corruption problem also has ramifications for the proposed TCP extensions for use on networks in which the product of bandwidth and round-trip delay is high (> 64k bytes). Specifically, the utility of Selective Acknowledgment (SACK) and Window Scaling is limited when using congestion control since the goals of these mechanisms are in direct conflict, as noted by RFC 1106. The SACK and Window Scaling options are designed to keep the pipe full of data, while the intention of congestion control is to allow the pipe to drain. However, while SACK and Window Scaling offer little benefit when congestion control is in effect, their use is not detrimental to performance in such a case, with the exception of the bit overhead of SACK.

D.2 CONGESTION CONTROL

In addition to the ability to enable or disable congestion control dynamically based on the source of packet loss, the SCPS extensions include modifications to the standard congestion control and avoidance algorithms. These changes consist of the implementation techniques proposed by TCP Vegas, and they do not involve any alteration to the TCP specification.

The SCPS TCP extensions also provide a rate control mechanism. This open-loop control system is similar to the leaky-bucket algorithm used for flow control of Available Bit-Rate (ABR) traffic in ATM [11]. It enforces an upper bound on the rate at which TCP can transmit to keep it from over-running the capacity of the link or the interface. A rate bound can be set by a network manager for each interface independently. This mechanism is especially important as a throttle when congestion control is disabled.

D.3 RETRANSMISSION AND ACKNOWLEDGMENT STRATEGY

In the absence of congestion, the best throughput performance can be obtained only when link idle time is minimized. Selective acknowledgments provide more information to the data-sender than TCP's cumulative acknowledgments and can aid in keeping the channel occupied. This is especially true in a unidirectional data transfer, as opposed to an interactive dialog in which the next data to be transmitted depends on the reply to the last data sent. In an unidirectional transfer of data, the optimal throughput is realized when the data-sender never needs to halt transmission waiting for an acknowledgment. As discussed previously, operating with a large enough window to satisfy the bandwidth-delay product of the network is one requirement for keeping the sender active. However, large windows alone are not sufficient to keep the pipe full. A prudent retransmission and acknowledgment strategy is also required to efficiently utilize the bandwidth when packet loss due to sources other than congestion is experienced.

When packet losses are a result of corruption, and congestion control is not activated by these losses, a selective acknowledgment strategy will improve TCP throughput performance. As discussed above, the SCPS TCP extensions provide a means for distinguishing congestion from noise, and as a consequence TCP is able to invoke its congestion control algorithms appropriately. The retransmission and acknowledgment strategy described in this section is most efficient when losses are not caused by congestion, and congestion control is not invoked; however, it operates independently of the congestion control mechanisms. This strategy is optimized to keep the pipe full and utilize the bandwidth as efficiently as possible within the constraints imposed by the effective window. As a result, the best throughput performance is obtained when the sender is not congestion-window limited in transmitting.

However, the congestion control logic may become active or inactive during the course of the connection without affecting the correct operation of the retransmission and acknowledgment strategy. In fact, if the network becomes congested, better performance will be obtained if the congestion control algorithms are invoked and the pipe is permitted to drain. The performance gains alluded to below will be realized in the case where losses are due to corruption, and this fact is recognized by TCP. When the losses are a result of congestion, the standard TCP congestion control and avoidance algorithms will govern the sender's throughput behavior, regardless of the retransmission and acknowledgment scheme employed.

TCP's cumulative acknowledgment mechanism provides limited information to the data-sender regarding which segments have successfully reached the destination. In the absence of complete information, the data-sending TCP must make assumptions about the state of the receiver's resequencing queue when retransmitting segments. The cumulative acknowledgment authoritatively tells the data-sender the highest sequence number that has successfully been received in-sequence. However, it provides no information about any segments that may have correctly been received out-of-sequence beyond the segment being ACKed. There are two opposing philosophies for handling this lack of information that lead to two distinct approaches for retransmission in reaction to a time-out or reception of multiple duplicate acknowledgments. These approaches are implementation details and are not governed by the TCP specification, and in practice, approaches that lie on the spectrum somewhere between these two extremes may be adopted. Note that Berkeley Standard Distribution (BSD) Unix Networking Software Release 2.0 follows the conservative approach described below [12].

1) The conservative approach dictates that when a retransmission is deemed appropriate, the sender retransmits *only* the oldest unacknowledged segment and no others. Once it retransmits this single segment, the sender may continue transmitting segments at the point where it left off before the retransmission, within the constraints of the effective window. The implicit assumption in this case is that only a single segment has been lost and the receiver has correctly received and queued all segments that were transmitted after the missing segment.

2) In contrast, the aggressive approach maintains that when a retransmission is necessary, the sender retransmits the oldest unacknowledged segment as well as every segment that it had sent after that segment. This "Go-Back-n" scheme operates under the assumption that the receiver has lost a succession of segments.

Both of these approaches have merit; however, each is optimized for a different operating environment. The conservative approach performs well when individual packets are sporadically lost. Here, unnecessary retransmissions of correctly received segments are avoided. In contrast, the aggressive scheme works well when sequences of consecutive packets are dropped. In the aggressive retransmission case, all of the lost segments are retransmitted without waiting for further information; although, it is imperative that such retransmissions be metered out slowly to avoid causing congestion. On the other hand, each retransmission approach performs poorly when faced with a loss profile that violates its assumptions. In the case of the conservative approach, when a string of consecutive segments is lost, only a single segment is retransmitted per round-trip time. Each time a segment is retransmitted, it is acknowledged; however the data-sender must wait for further notification (in the form of duplicate ACKs or a time-out) to discover the next lost segment. This strategy recovers very slowly from a sequence of losses. Conversely, the aggressive approach performs poorly when only a single segment is lost. In this case, bandwidth is wasted as a window's worth of segments are retransmitted when only a single retransmission was actually necessary.

By telling the sender exactly which segments have been successfully received (or which segments are missing), a selective acknowledgment strategy can provide better performance than the simple cumulative acknowledgment scheme of TCP, regardless of which retransmission approach is employed. (Again, we emphasize that the performance gain will be realized only when losses are not due to congestion, and TCP does not invoke a congestion control response when loss occurs.) This performance improvement, in terms of a reduced number of retransmissions, fewer retransmission time-outs, and higher throughput, can be more substantial in a noisy environment with a high bandwidth-delay product. The SCPS developers analyzed the RFC 1072 [13] SACK option and the RFC 1106 NAK option and borrowed ideas from them in designing the SCPS Selective Negative Acknowledgment (SNACK) option. Refer to RFC 1072 and RFC for a detailed discussion of these SACK and NAK, respectively; an analysis of these options with respect to the SCPS requirements follows.

The ability to acknowledge multiple discontiguous blocks is desirable, and is provided by the Selective Acknowledgment mechanism proposed in RFC 1072; however, three problems exist with the scheme:

1) The SACK option has limited coverage. Because an upper bound exists on the size of the TCP header (60 bytes, of which 20 bytes are consumed by the standard TCP header), and because a SACK option that specifies n blocks has a length of $4n + 2$ bytes, a single SACK option is capable of specifying at most 9 blocks. Moreover, if other TCP options are in use, which compete for space in the TCP header, a given SACK option may not be able to specify even 9 blocks. This limitation reduces the advantage gained by using Selective Acknowledgments rather than cumulative acknowledgments, especially when operating with large windows in a noisy environment.

2) The SACK option is imprecise. When RFC 1323 [14] Window Scaling is enabled, the window size can effectively be as large as a 30-bit quantity. The SACK option uses a 16-bit field to identify the location, or Relative Origin, of a block of data within this window

space. Likewise, the Block Size field, which specifies the size in bytes of a contiguous block of data is a 16-bit quantity. Clearly, these fields cannot address the entire window space when scaling is enabled. The two solutions to this problem proposed in RFC 1072 are both inadequate. The first is to expand the SACK Relative Origin and Block Size fields to 24 or 32 bits each. From the perspective of bit-efficiency, this solution is unacceptable. In addition, this approach drastically reduces the number of blocks that can be specified by a SACK option. The second proposal is to scale the SACK fields by the same value as the window. This solution introduces imprecision into the acknowledgment, since the SACK option must report block offsets and lengths in multiples of the window scale factor, which may not be a multiple of the segment size. To reconcile this imprecision, it is necessary to adopt a conservative approach and unnecessarily retransmit some data when there is doubt as to which segments are being acknowledged. This approach also makes inefficient use of the channel.

3) The SACK option is incompletely specified. RFC 1072 describes the format of the SACK option, but it does not cover other essential issues, such as when to send a SACK as the receiver of data, or how to process one as the data sender. RFC 1072 also fails to mention the interaction between the SACK option and the congestion control algorithms, which governs the system throughput in the event of packet loss in standard TCP.

While the RFC 1106 NAK option is reasonably bit-efficient, it has the ability to signal only a single “hole” that exists in the sequence space in the receiver's buffer. A more powerful mechanism, capable of specifying multiple holes is desirable in the noisy, long-delay environment to provide the sender with more complete information about the state of the receiver's potentially large out-of-sequence queue. (Note that large send and receive buffers are required when operating with large windows.)

The SCPS Selective Negative Acknowledgment (SNACK) is a variable-length TCP option that has the ability to signal the presence of multiple holes in the receiver's resequencing queue in a bit-efficient manner. The SNACK option consists of 5 fields: the kind and length fields required of all TCP options, followed by the “Offset” and "Hole1" fields (each 16 bits long) and an optional variable-length bit-vector. Offset specifies the displacement from the ACK number carried in the regular TCP header to the starting location of the first hole that is being signaled by this particular SNACK option. The Hole1 field specifies the size of this hole. Note that this hole is *not* necessarily the first hole in the receiver's overall out-of-sequence queue. Both the Offset and Hole1 values are expressed in Maximum Segment Size (MSS) units. The bit-vector then signals zero or more additional holes, also expressed in terms of MSS-sized blocks. The bit-vector maps the sequence space of the receiver's buffer beginning one byte beyond the end of the block specified by Hole1. Each “0” in the bit-vector signifies that one or more bytes are missing in the corresponding MSS-size block of the receiver's resequencing queue. (Note that any zeroes to the right of the rightmost “1” in the bit-vector are *not* interpreted as a hole.) The length of the bit-vector, which may vary at the SNACK-sender's discretion, is determined from the option length. Use of the SNACK option is enabled by another option, SNACK_OK, which may be sent only on the SYN segment of the connection.

While an out-of-sequence queue exists, the data-receiver scans its receive buffer, forming SNACK options and sending them on outgoing ACK segments. By setting the Offset field to zero, the data-receiver can NAK a block beginning at the ACK number in the TCP header. Alternatively, by specifying a non-zero Offset value, the SNACK option can begin by addressing any arbitrary portion of the sequence space. The latter capability is especially useful when the out-of-sequence queue is large, even in terms of MSS units, and a single SNACK option is unable to reference the entire sequence space because of the limit on the TCP header size. In such a case, multiple SNACK options can be sent in which each continues specifying holes in the receive buffer where the last left off.

At the data-receiver, there are considerations regarding whether transmission of the SNACK option should be delayed, and by how much, in anticipation of the missing segment(s) arriving out of order. This decision is implementation specific and should take into account the probability of segment misordering by the underlying network(s). Unless segment misordering is highly unlikely, delaying transmission of the SNACK option may offer some benefit. This assertion holds because the SNACK option forces immediate retransmission. For this reason, the SNACK-sender should be as certain as possible that retransmission in required. In the SCPS environment, acknowledgments are typically delayed substantially anyway because the uplink bandwidth (the acknowledgment channel) is severely limited in comparison to the downlink. Thus, ACKs are aggregated as much as possible, and in many cases, an ACK is sent just once per round-trip time.

As mentioned above, upon receipt of a SNACK option, the data-sender retransmits all segments necessary to fill the signaled holes. This behavior is similar to that dictated by the RFC 1106 NAK option, in contrast to the RFC 1072 SACK recommendation. Receipt of an RFC 1072 SACK requires the data-sender to simply mark the selectively ACKed segments as acknowledged, but does not cause retransmission of the segments that were previously sent but not acknowledged by the SACK. The more aggressive retransmission scheme adopted here is appropriate when the goal is to prevent retransmission time-outs, which cost more in terms of link idle time than unnecessary retransmissions cost in terms of wasted bandwidth. The data-sender may process the SNACK option by first retransmitting the segments that fill the gap corresponding to Hole1. If a bit-vector is present, the data-sender may easily process it by left-shifting the bit-vector until the last “1” is shifted out while retransmitting the segment corresponding to each “0” encountered in the process.

Finally, note that large buffers are required when trying to keep the pipe full using a selective acknowledgment scheme. To prevent the pipe from emptying while responding to SACKs, the retransmission buffer must be large enough to store the amount of data that can be transmitted in more than 2 RTTs. This interval is determined by the time between the initial transmission of a segment that is subsequently lost, and its eventual acknowledgment over 2 round-trip times later, with an intervening (possibly delayed) SNACK and a retransmission 1 RTT after the initial transmission.

D.4 WINDOW SCALING

The SCPS extensions include the TCP Window Scaling option exactly as specified in RFC 1323.

D.5 ROUND-TRIP TIME MEASUREMENT

The SCPS extensions include both the RFC 1323 Timestamps option and the TCP Vegas mechanism for accurately measuring the round-trip time. One of these two mechanisms, but not both, is chosen for use on a particular connection.

D.6 PROTECTION AGAINST WRAPPED SEQUENCE NUMBERS

The SCPS extensions include the Protection Against Wrapped Sequence Numbers (PAWS) mechanism as specified in RFC 1323.

D.7 BEST-EFFORT TRANSPORT SERVICE (BETS)

TCP offers only a single, fully-reliable service while UDP offers only an unreliable service. Some applications that are used in the SCPS environment require an intermediate level of reliability. The Best Effort Transport Service (BETS) option provides such a data transfer service that guarantees uncorrupted and in-sequence data delivery, but possibly with gaps. In the terminology introduced in RFC 1693 [15], BETS provides a partially reliable, ordered service.

When BETS is enabled, the data-sending application has the ability to set thresholds of buffer utilization and retransmission count. Once one of these thresholds is exceeded, unacknowledged data is ignored and transmission of new data continues. The effect of BETS on the data-transmitter is that the application is never “blocked” indefinitely waiting for acknowledgments.

At the receiver, the application can specify maximum blocking thresholds in terms of buffer capacity and time. When one of these thresholds is exceeded due to an unfilled hole in the received data, the application is informed of the size of the gap (in octets) and is then delivered the data that is available. The effect of BETS is that the data-receiver is not blocked indefinitely waiting to fill “gaps” in the received data.

BETS supports three major types of operation:

1) High-speed data transfers of repeated data (such as images), for which the loss of a portion of the data (i.e., a portion of a scan line) is important only in that it might cause synchronization to be lost.

2) Reliable transfer of data in a buffer-limited environment, for which availability of new data is more important than retransmissions of old data, if buffers are exhausted.

3) Highly-reliable operation when no acknowledgment channel is available.

Missions with some or all of these types of communication scenarios should benefit from the Best Effort Transport Service.

D.8 SCPS TP HEADER COMPRESSION

SCPS TP Header Compression is to be used on connections that require high bit-efficiency. The requirement for high bit-efficiency may result from the presence of low-data rate links in one or both directions of the communication path, or from high utilization of those links. For most space missions, link bandwidth is considered a scarce resource, and the overhead of TCP segment headers is considered too high.

A significant amount of work has been done outside the SCPS activity to reduce the overhead of TCP/IP headers. This work is documented in RFC 1144. The compression specified in RFC 1144 is intended for use on low-speed serial links, and addresses the problems of maintaining interactive response for programs such as telnet. RFC 1144 header compression operates at the link layer. The link layer maintains connection state tables for inbound and outbound TCP/IP connections. The first header for each connection seen by the link layer is not compressed. Subsequent headers are encoded by sending only the fields that changed from the previous header. Additionally, the sequence number, acknowledgment number, urgent pointer, and window values are encoded as *changes* to the previous value. At the receiving side, an uncompressed TCP/IP header is created by applying the changes to the saved header to create a new TCP/IP header. This new header is saved in the connection state table and is forwarded to the destination along with the data. Since the information in the compression state tables will be corrupted if a segment is lost or damaged (misordering is typically not a problem at the link layer), the (unmodified) TCP checksum is included in all TCP segments. If the decompression state is corrupted, the TCP checksum will fail at the receiving TCP endpoint and the corrupted segment will be discarded. Retransmissions are forwarded uncompressed by the RFC 1144 compressor, and are used to resynchronize the decompressor's state. (Note that if a segment is lost or corrupted, *all segments following it will be decompressed improperly, causing them to be discarded by the receiving TCP endpoint.* This behavior continues until the sending TCP entity retransmits the lost segment(s), resynchronizing the compressor and decompressor. This is typically not a problem for RFC 1144 operation, since it is designed primarily for interactive operation in which there are typically only a few octets of data outstanding at one time.)

While RFC 1144’s method of header compression works well in low bandwidth-delay networks that have stable connectivity and low bit-error rates, it is poorly suited for the SCPS environment. The high bandwidth-delay products mean that any failure of the decompression will result in significant data loss. Further, the possibilities of high bit-error rates or changing topologies or both increase the probability that decompression failure would happen more often in the SCPS environment than in the terrestrial environment.

SCPS TP header compression draws on ideas developed in RFC 1144 and previous header compression schemes. Unlike RFC 1144, SCPS TP's compression operates at the transport layer, in an end-to-end manner. Therefore, it is unaffected by changing network connectivity. RFC 1144's decision to send deltas from the previous value when the value changes makes it particularly susceptible to decompression failure if loss or corruption is experienced. Rather than sending deltas, SCPS TP sends the full field value. This results in slightly less per-packet efficiency, but allows resequencing to be successfully accomplished, eliminating the go-back-n retransmission behavior that RFC 1144 imposes. Further, SCPS TP sends sequence number fields whenever data is present (whether the field changed from the previous value or not), and always sends the window field when an acknowledgment field is sent. These modifications improve the robustness of SCPS TP compression, at the expense of some efficiency. SCPS TP allows certain options to be compressed - specifically, the Selective Negative Acknowledgment option (SNACK), and the TCP Timestamps option. These may occur frequently, and are parsed and compressed by the header compression software. Other TCP options are included uncompressed. Finally, since the SCPS TP compressed packets are essentially "stand-alone", the checksum covers the compressed packet header and the user data, not the uncompressed header and the user data.

D.9 OTHER SCPS EXTENSIONS

The SCPS extensions include additional features that are not of significant relevance to this experiment, such as a record boundary option and a priority mechanism. For further discussion of these extensions, see [5].

APPENDIX E: SCPS TP IMPLEMENTATION BACKGROUND

The TCP/IP protocols are typically implemented in the kernel of the Unix operating system and are accessed by applications through system calls. The kernel is a logical place for networking protocols to reside from both the conceptual and practical points of view. The protocols simultaneously provide services to multiple processes that reside outside the kernel, and they can operate more efficiently as kernel routines without the overhead and scheduling constraints associated with being a Unix process. However, there are disadvantages to building a prototype that resides within the Unix kernel. Development, debugging, and testing of kernel implementations is significantly more difficult and time-consuming than that of application programs. Also, kernel implementations are closely bound to the specific version of the operating system for which they are designed, as well as to the platform on which that version of the operating system runs. A final, practical issue is that to produce a kernel implementation of a protocol, a developer requires access to the kernel source code, which is prohibitively expensive for some of the popular commercial operating systems (e.g., SunOS).

In designing the TCP prototype for the SCPS project, the advantages of working outside the Unix kernel prevailed. A primary concern for the SCPS project was portability of the prototype code, which dictated a user-space implementation. In fact, the SCPS protocols are expected to be hosted on satellites that do not have a multi-purpose operating system into which the protocols could be integrated, so developing the prototype as a stand-alone program is beneficial. Another major concern of the SCPS project was the cost of the prototype in terms of development time. Here again, a user process implementation was the obvious choice. Consequently, the approach chosen by the SCPS development team was to implement an enhanced TCP prototype as a Unix application program.

One possible development choice was to extract the TCP/IP implementation from the Berkeley Unix source code and modify it to run as a user process. However, this option was not selected because the networking code is so intimately tied to the rest of the Unix operating system (e.g., the file and memory management systems) and is so difficult to dislodge from the kernel. Instead, a minimal implementation of TCP, called TinyTCP [16], that already was structured to run as a stand-alone program was chosen as the starting point for the prototype implementation. TinyTCP lacks many of the basic features required of an RFC 1122 [17] compliant TCP implementation; however, the philosophy was adopted that it is easier to add features to a minimal implementation than remove them from a full-featured implementation. Among the features missing from the original version of TinyTCP are the following:

- Window-based flow control
- Retransmission buffers
- Resequencing buffers

- An Application Programming Interface (API)
- Congestion control and avoidance mechanisms
- Round-trip time estimation
- Delayed acknowledgments
- Nagle's algorithm
- Karn's algorithm
- Sender and receiver Silly Window Syndrome (SWS) avoidance algorithms

The SCPS team added the above features, as well as the SCPS extensions, to the TinyTCP implementation in transforming it into the SCPS-TP prototype. They also made other fundamental changes to TinyTCP. First, we ported TinyTCP from its native environment to run as user-level Unix process over raw IP sockets on any Berkeley-derived Unix host. Next, we restructured the control flow of the program. When TinyTCP was originally ported to the Unix environment, the transport protocol and the application using its services, either the data-sender or data-receiver, were compiled and linked together into a single program and run as a Unix process. The transport protocol served as the scheduler for its application "process," by calling an application-supplied function when necessary. For example, when TinyTCP was ready to pass received data to the application, it invoked the application's data-consumer function. In building the SCPS-TP prototype, we replaced this control structure in favor of the more generic and flexible approach of lightweight threads and a thread scheduler.

Threads are low-overhead "processes" that execute in user space. Threads each maintain their own private variables and state information, but as parts of a single Unix process, they also share globally-accessible process state, including memory. The thread scheduler acts just like the process scheduler in a multi-tasking operating system that provides context-switching capability. It removes the currently running thread from the execution state, places it in the run queue, and gives control of the CPU to the next runnable thread in the queue. The SCPS thread-scheduler is non-preemptive; it does not decide when a context switch should occur and interrupt the currently executing thread asynchronously. Instead, each thread has the responsibility to actively relinquish control of the processor regularly by explicitly invoking the thread scheduler through a function call. Under this architecture, the transport protocol and the application are each implemented as separate threads, which are executed in a round-robin fashion by the thread scheduler. The thread scheduler, the application, and the transport protocol are all compiled and linked together into a single executable program.

The SCPS team has developed two prototype applications for use with the SCPS-TP protocol prototype, a data-sender and a data-consumer. Note that typical Unix TCP applications would require modification to run over the SCPS-TP prototype. While a socket-like API has been added to the SCPS-TP prototype, the semantics (as well as the syntax) of the service calls differ from those of Berkeley sockets. More significantly, every application that uses the SCPS-TP prototype must be structured as a thread that is linked in with the prototype. As such, an application must contain calls to the thread scheduler at periodic intervals. Consequently, popular TCP applications, like FTP, are not readily usable with the SCPS-TP prototype. Instead, a simple data-transmitting client (scps_init) and a data-receiving server (scps_resp) have been developed for use in debugging, testing, and evaluating the prototype.

GLOSSARY

ABR	Available Bit-Rate
AFMC	Air Force Materiel Command
AFSC	Air Force Systems Command
AOS	Acquisition of Signal
BER	Bit Error Rate
BERT	BER Tests
BRL	Ballistics Research Laboratory
DOD	Department of Defense
ED	Experiment Director
GMT	Greenwich Mean Time
Hr.	hour
HSIS	High-Speed Interface S-Bus
IP	Internet Protocol
IRON	Inter-Range Operation Number
JPL	Jet Propulsion Laboratory
Kbps	Kilo bits per second
Kbyte	Kilo byte
LAN	Local Antenna Network
LOS	Loss of Signal
MSS	Maximum Segment Size
min.	minute
N/A	Not applicable
NASA	National Aeronautics & Space Administration
PAWS	Protection Against Wrapped Sequence Numbers
PST	Pacific Standard Time
SCPS	Space Communications Protocol Standards

SCPS-FP	SCPS File Protocol
SCPS-NP	SCPS Network Protocol
SCPS-SP	SCPS Security Protocol
SCPS-TP	SCPS Transport Protocol
SCPS-TWG	SCPS Technical Working Group
SGLS	Space Ground Link System
SNACK	Selective Negative Acknowledgment
STGS	S-Band Transportable Ground Station
SWS	Silly Window Syndrome
TBD	To be determined
TC	Test Conductor
TCP	Transmission Control Protocol
TSTR	Transportable Space Test & Evaluation Resource
UDP	User Datagram Protocol
U.S.	United States
USSPACECOM	United States Space Command

DISTRIBUTION LIST

INTERNAL

W159
R. C. Durst (10)
G. S. Miller
M. J. Zukoski

D048
D. E. Ernst
B. A. Feldmeyer
J. G. McIlnay
R. Ramirez
E. J. Travis (Gemini Industries),
c/o MITRE Reston M/S W650

PROJECT

USSPACECOM, J4S
Attn: Capt. Syndergaard
250 S. Peterson Blvd.
Peterson Air Force Base
Colorado Springs, CO 80914-3310
(Original + 4 copies + disk in DOS format)

EXTERNAL

SMC/TEO
Attn: Col Daryl Joseph
P.O. Box 007
Onizuka AS
1080 Lockheed Way
Sunnyvale, CA 94089-5000

Mr. Adrian Hooke (5 + disk)
NASA-JPL
M/S 301-235
4800 Oak Grove Drive
Pasadena, CA 91109-8099

Mr. Nick Shave
Space Systems
Defence Research Agency
R16 Bldg.
Farnborough, Hampshire
GU146TD, U.K.

Ms. Elaine Skrzypczak
SAIC
8301 Greensboro Drive
Suite 1200
M/S E-12-1
McLean, VA 22102

Mr. John Rush (5 + disk)
NASA/HQ
300 E. Street S.W.
Washington, DC

Mr. Howard Weiss
SPARTA, Inc.
9861 Broken Land Pkwy
Suite 300, Columbia, MD 20706

Mr. Tom Bleier (5)
The Aerospace Corporation
1320 Orleans Drive
Sunnyvale, CA 94089-1135

Ms. Lori Jeromin
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02173-9108

Ms. Mary Joan Trafton
The Aerospace Corporation
P.O. Box 92957 M5/500
Los Angeles, CA 90009-2957

TEO/DS
Attn: Mr. A. Reagan/J. Hansen
Loral
P.O. Box 007
Onizuka AS
1080 Lockheed Way
Sunnyvale, CA 94089-5000

TEO/DS
Attn: Mr. A. Reagan/J. Hansen
Loral
P.O. Box 007
Onizuka AS
1080 Lockheed Way
Sunnyvale, CA 94089-5000