4-2006

A Comparative Analysis of Transmission Control Protocol Improvement Techniques over Space-Based Transmission Media

Joseph M. Lawson

A COMPARATIVE ANALYSIS OF

TRANSMISSION CONTROL PROTOCOL

IMPROVEMENT TECHNIQUES OVER

SPACE-BASED TRANSMISSION MEDIA

THESIS

Joseph M. Lawson, Captain, USAF

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GIR/ENV/06M-08

A COMPARATIVE ANALYSIS OF TRANSMISSION CONTROL PROTOCOL IMPROVEMENT TECHNIQUES OVER SPACE-BASED TRANSMISSION MEDIA

THESIS

Presented to the Faculty

Department of Systems and Engineering Management

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Information Resource Management

Joseph M. Lawson, BSEE

Captain, USAF

March 2006

AFIT/GIR/ENV/06M-08

A COMPARATIVE ANALYSIS OF TRANSMISSION CONTROL PROTOCOL IMPROVEMENT TECHNIQUES OVER SPACE-BASED TRANSMISSION MEDIA

Joseph M. Lawson, BSEE
Captain, USAF

Approved:

/signed/ 14 April 2005
Dr. Michael R. Grimaila (Chairman) Date

/signed/ 14 April 2005
Dr. Alan R. Heminger (Member) Date

/signed/ 17 April 2005
Dr. Dennis D. Strouble (Member) Date

AFIT/GIR/ENV/06M-08

Abstract

The Internet has revolutionized the way information is shared between geographically separated entities and has become the medium of choice for data sharing for both commercial and military units. This reality seemingly simplifies network design, but characteristics of the underlying transport protocols require consideration when used in a high delay, high error environment, such as a satellite transmission architecture. Of great interest are modifications to the Transmission Control Protocol (TCP) that enhance TCP performance in these poor conditions. Although extensive research has been conducted concerning TCP optimization, few have assessed the performance benefit a given modification can provide.

The purpose of this study was to assess the throughput improvement afforded by the various TCP optimization techniques, with respect to a simulated geosynchronous satellite system, in order to provide a cost justification for the implementation of a given enhancement technique. It was determined that each technique studied, which included the Space Communication Protocol Standard – Transport Protocol (SCPS-TP), window scale, selective acknowledgements (SACKs), and combinational use of the window scale and SACK mechanisms, provided varying levels of improvement as compared to a standard TCP implementation. In terms of throughput, SCPS-TP provided the greatest overall improvement, with window scale and window scale/SACK techniques providing significant benefits at low levels of bit error rate (BER). The SACK modification improved throughput performance at high levels of BER, but performed at levels comparable to standard TCP during scenarios with lower BER levels.

Acknowledgements

There are many people that I owe a multitude of thanks for assistance in the completion of this thesis. First and foremost, I would like to express gratitude to my wife and my two sons for their love, support, and understanding during the arduous thesis process. My parents, as well as my brother, have also provided a tremendous amount of support during my time at AFIT. In addition would also like to thank my classmates, in particular Steve, Bob, Scott, Mark, and Pat, who have been there for me in times of need and were some of the finest people I have ever served with. Finally, I need to express thanks to my advisor, Dr. Michael Grimaila, for his unwavering support, technical insight, and professionalism.

Joseph M. Lawson

Table of Contents

Page

List of Figures

List of Tables

Page

A COMPARATIVE ANALYSIS OF TRANSMISSION CONTROL PROTOCOL IMPROVEMENT TECHNIQUES OVER SPACE-BASED TRANSMISSION MEDIA

I. Introduction

Background

Leasing bandwidth on commercial geostationary (GEO) satellites is a costly endeavor, with prices ranging up to $1.5M for an 8 Mbit/sec allocation per year. In 2004, the Department of Defense (DoD) spent nearly $500M for commercial satellite transponder/channel leases (Dykewicz, 2005). In addition, military supported constellations, particularly those spacecraft that are part of the Defense Satellite Communications System (DSCS) and Wideband Gapfiller programs (which currently or will support a majority of the United States military's X-band communication requirements), require significant DoD funding to maintain required operational and maintenance levels. As a result, military communication support entities, as a consumer of both commercial and military satellite systems, are continually seeking ways to efficiently utilize space-based communication bandwidth allocations in order to maximize return on investment. Unfortunately, commonly used data communication network protocols, particularly the Transport Control Protocol (TCP) piece of TCP/Internet Protocol (TCP/IP) suite, have limited efficiency in a satellite-based transmission system. This is primarily due to the negative relationship between satellite link traits (high latency and bit error rates) and congestion control algorithms employed by TCP to detect and react to transmission errors and faults. To counter this paradigm both military and commercial organizations have relied upon use of Performance Enhancing Proxies (PEPs) to maximize bandwidth usage.

PEPs have a relatively short history of use in US military operations. The first large scale employment of these devices by the Department of Defense occurred during the initial stages of Operation IRAQI FREEDOM at the Landstuhl Standardized Tactical Entry Point (STEP) in Landstuhl, Germany. STEP sites represent communication elements that provide deployed combat units access (typically via satellite or terrestrial mediums) to the Defense Information Systems Network (DISN) and DISN's associated services (secure/non-secure Internet circuits, video teleconference circuits, messaging systems, secure/non-secure voice circuits, etc.). During IRAQI FREEDOM, an agreement was reached between the STEP program manager, Joint Staff J6, and 5th Signal Command for the installation of 5th Signal Command provided Mentat SkyX PEPs at the Landstuhl STEP site in order to improve satellite bandwidth utilization between the STEP site and US Army V Corps' Logistic Support Areas in Iraq and Kuwait. Though the initial planning and installation phases of the supplied PEPs were somewhat problematic, the subsequent performance benefits of these devices provided a great bit of incentive by the STEP program office for the procurement of additional PEPs for installation at all 18 active STEP sites worldwide.

Procurement and installation of PEPs at STEP sites worldwide was rolled into the ongoing Enhanced STEP upgrade program. Depending on the "size" (termed as either single or dual, which was typically based on the number of apertures available at each site) of a STEP site, the type and number of upgrades afforded to it varied. In general, with regard to PEPs, a dual STEP site would receive 32 devices for use on the Non-secure Internet Protocol Router Network (NIPRNET) circuits and the Secure Internet Protocol Router Network circuits (SIPRNET) (64 devices total). In contrast, a single STEP site would receive a total of 32 PEPs for use on both NIPRNET and SIPRNET circuits. Though PEPs were viewed to be effective in mitigating TCP link degradations associated with satellite communications, the additional burden of support (training, maintenance, and operation) and the cost of acquisition for these devices proved to be a significant undertaking.

The average cost of a PEP procured by the STEP program was $3,000, with training (train the trainer program) costs varying between $10,000 and $15,000 per person. Taking in consideration that 18 STEP sites were to be upgraded, with a minimum of 32 devices, as well as the training of at least 18 personnel, the cost for the PEP implementation of this magnitude could easily exceed $2M. In addition, STEP sites were also expected to allocate a large amount of manpower in the reorganization of their sites to accommodate the rack space and power requirements of these devices. Some STEP sites, such as Ramstein STEP (located at Ramstein Air Force Base, Germany) had severe site limitation both with power availability and physical space available. In addition, STEP sites also needed to reevaluate their respective facility back-up power mechanisms; assessments would provide insight as to whether or not facility back-ups could have handled the additional load presented by the PEPs in the event of a power outage. If the power back-up system could not handle the load, acquisitions would need to be made to further robust the system.

Due to the acquisition, installation, and training costs for PEPs it is not surprising that many organizations are researching techniques to incorporate in commonly employed technologies (routers, satellite modems, etc.) to combat link performance issues and negate the need for another device to acquire, maintain, and operate. Of particular interest is the use of selective acknowledgements (SACKs) and increases in the TCP window size, that have been incorporated in already released Cisco router Internetwork Operating System versions. Since Cisco products represent the largest population of network devices used by military communication units, it is reasonable to assume that a significant cost savings (in terms of dollars and physical requirements) could be realized if SACKs and window scale techniques rival the performance benefits of a PEP system.

Problem Statement

Though PEPs have been proven to provide measurable benefits with respect to TCP throughput in a high delay/error transmission environment, the cost of acquisition, needed support mechanisms, and facility requirements necessitate the need to find alternatives for organizations that cannot fund, support, or have the facility capacity required for PEP implementation. Though these organizations cannot support a PEP solution, they may still have the need to maximize their respective transmission system throughput capabilities and, as such, require some type, or a combination, of TCP enhancement technique. The research presented in this paper will investigate the performance benefits of SACKs and window scale strategies in a satellite transmission system and contrast these findings to those obtained from a PEP-enhanced solution. Through statistical analysis (Analysis of Variance, Tukey analysis, and descriptive statistics) observations can be made concerning the differences (if any) in performance (in terms of throughput) between the respective enhancement strategies.

Research Questions

Based on the problem identification of the previous section, it can be seen that a number of research questions must be formulated and answered in order to provide adequate information to military units concerning the performance gap (or lack thereof) between multiple types of TCP enhancement strategies. The research questions for this study can be seen below.

RQ1). What is the performance benefit (in terms of throughput) of implementing a SACK enhancement to TCP when TCP is utilized over a satellite transmission system?

RQ2). What is the performance benefit of implementing a window scale modification to TCP when TCP is utilized over a satellite transmission system?

RQ3). What is the performance benefit of implementing both a SACK enhancement and a window scale modification to TCP when TCP is utilized over a satellite transmission system?

RQ4). What is the performance benefit of utilizing PEPs to improve TCP sessions that occur over a satellite transmission system?

RQ5). Are there statistical differences between the respective TCP enhancements? If so, how can costs (in terms of acquisition and support requirements) be expressed as a decision point for execution of a given TCP enhancement?

Implications

The potential implications of this study, based on experimental findings, can allow military units to weigh the benefits of differing TCP enhancement techniques in order to select the methodology that best fits their transmission requirements. Units who value minimization of equipment footprint (such as a combat communications element), may have limited funding, or are unwilling to support additional equipment may use this study to determine if performance provided by TCP enhancements, that can be implemented on already in-place equipment, can provide a suitable cost/benefit ratio. On the other hand, these same units may use this study to provide justification for the additional cost (in terms of acquisition and support) in order to gain greater transmission performance afforded by a PEP-based solution.

Scope

The scope of this research encompasses the use of TCP/IP-based services (such as voice-over-IP, IP-based video teleconferencing, Internet access, etc.) via GEO satellite transmission systems. This type of communication system is commonly used by deployable military communication units, as well as fixed military installations (such as STEP/Teleport sites) that communicate with down range elements and other inter- and intra-theater fixed-based organizations. In addition, this study also has applicability to commercial entities that transmit TCP/IP-based services through GEO satellite systems.

Limitations

Access to satellite resources, such as access time on a Defense Satellite Communications System asset, would require a considerable amount of flexibility with respect to time and availability due to on-going military operations and operation at a research level of precedence. Due to this fact, it will be necessary to simulate network traffic and the satellite environment in order to ascertain the benefits of various TCP enhancements. Though a more manageable approach, it is fraught with limitations. Obviously, without use of an actual satellite system, it will be impossible to simulate all environmental factors. In addition, not all of the equipment that is part of a military satellite transmission system (such as crypto gear, satellite modems, up/down converters, etc.) will be modeled and as a result, the research model may not adequately cover system idiosyncrasies (such as processing delay associated with multiplexing, cryptography equipment, etc.) encountered by real world transmission systems. Though the research model may not be all encompassing, a majority of transmission factors will be represented and should provide a solid base of knowledge in which other factors can be added later to further robust this study and more accurately predict system outcomes.

Document Overview

This chapter provided a brief background on the deficiencies of TCP in a high delay and error environment and covers some methods used to counter these deficiencies. Chapter One also presents numerous research questions, most of which revolves around comparing the effectiveness of TCP enhancement methods, of which one method requires an external device whereas the other methods can be incorporated in many of the existing router/satellite-based communication topologies.

The remainder of the thesis is structured as follows. Chapter Two provides a detailed literature review of peer reviewed journals, books, and military publications related to TCP mechanisms, TCP performance deficiencies when used over high delay and error environments, and various enhancement methodologies used to improve TCP performance in a satellite-based, network environment. Chapter Three covers the experimental design and methodology. Chapter Four will give a summary of findings from the experimentation and subsequent analysis. Chapter Five will present the research conclusions and recommendations for further research.

II. Literature Review

Chapter Overview

This chapter will present an overview of the TCP/IP protocol suite, to include the algorithms utilized to overcome network errors and congestion. In addition, a review of TCP/IP performance impact variables, namely the satellite environment, transmission link BERs, and bandwidth asymmetries will be presented. Next, mechanisms that can be incorporated in the TCP algorithms (such as use of selective acknowledgements and an increased window size) to mitigate issues with propagation delay and high BER will be discussed. Next, the use and benefits of performance enhancing proxies (PEP) will be presented. Finally, the Space Communication Protocol Standard–Transport Protocol (SCPS-TP), used by many PEP manufactures, will be discussed.

TCP/IP Overview

IP represents the preferred method for the logical mapping of devices connected via the Internet. IP is able to resolve the logical location of every device on the Internet through the assignment of a global IP address to network devices and with the utilization of network routing tables that record IP address assignments (Shaughnessy, 2000:81). Through this system, IP datagrams traveling through a network contain the necessary information (contained in the packet header) that allows IP to resolve the best path that must be taken in order to reach the intended network destination (Lammle, 2000:117). IP is a connectionless protocol that contains no mechanism to ensure the delivery of a data packet (Held, 1999:452). IP does, however, guarantee that data packets will not route indefinitely in a network through the use of a Time-to-Live (TTL) field in the packet header. The TTL field contains the maximum time (based on a hop count, where datagram passage through a router or other network node will reduce the TTL field by one; the maximum value the TTL field can be set to is 255) that a data packet can exist in a network (Feit, 1993:99; Held, 1999:465). Once the TTL value is exceeded, the network discards the data packet (Feit, 1993:99; Held, 1999:465). In order to guarantee the reliability of data packet delivery in a network, a secondary protocol, such as TCP, must be utilized.

TCP is a connection-oriented protocol that validates transmission of data segments via a positive acknowledgement system (Feit, 1993:179; Postel, 1981:1). When a networked computer transmits a data block, TCP breaks the block into segments and assigns each individual segment a sequence number (Feit, 1993:179; Postel, 1981:10). A segment is defined as a grouping of a stream or multiple streams of eight binary digits. Segments can have variable lengths, with the maximum length (termed maximum segment size) agreed upon by the sender and receiver during the initial establishment of a TCP session (Feit, 1993:188; Postel, 1981:42). The assignment of sequence numbers to segments enables the receiving end of the transmission to arrange segments back to the original order, enabling the reconstruction of the transmitted data block if packets are received out of the original order due transmission issues such as loss (and subsequent segment retransmission) or use of differing transmission paths due to network congestion (Lammle, 2000:107). In addition, the receiver of a data transmission transmits an acknowledgment (ACK) to the sender that specifies which data segments have arrived. Each ACK contains the sequence number of the next anticipated segment of a data transmission (Allman, 1997:2). Figure 1 provides an example of the ACK system behavior of TCP from the perspective of a receiver.

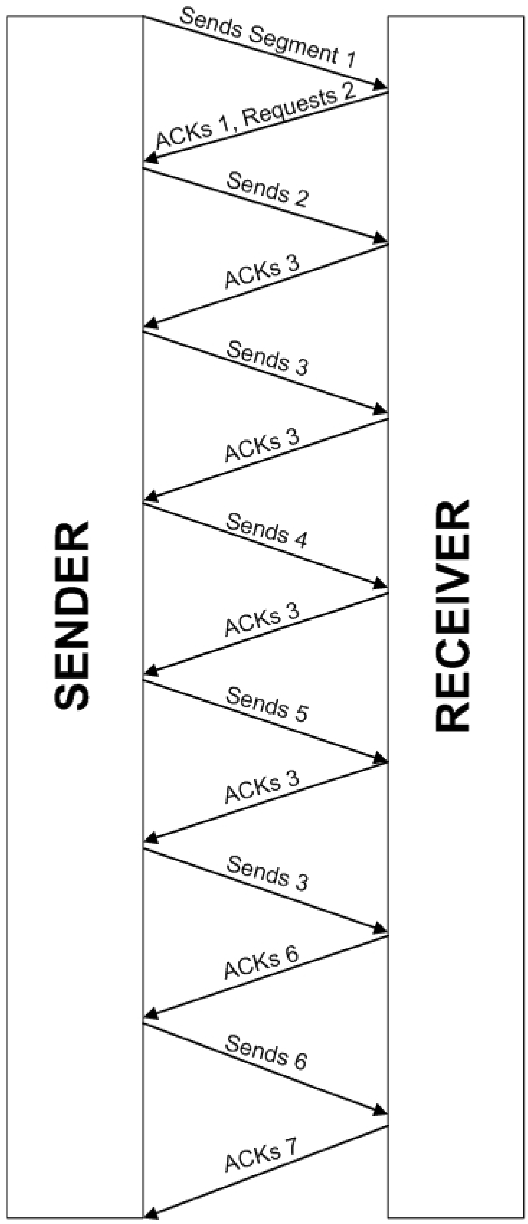

Figure 1. TCP Acknowledgement Example

From the above graphic, the receiver verifies receipt of segment one with the transmission of an ACK that contains sequence number two. The behavior repeats with the receipt of segment two, as the receiver transmits an ACK with sequence number three. The receiver then receives segment four as opposed to the expected segment three. As a response, the receiver validates the receipt of segment four with an ACK containing segment number three. This behavior will continue until segment three has been received. As can be seen in the example, segment three is received after segment five, and the receiver continues to transmit an ACK with the sequence number for the next expected segment.

The ACK system also ensures segment delivery via the inclusion of a retransmit timeout (RTO). The RTO represents the amount of time the originator of a data block will wait for an ACK of a segment's receipt before the segment will be retransmitted (Feit, 1993:200; Allman, 1997:2). RTO can be calculated utilizing the equation (Jacobson and Karels, 1988: 6; Karn and Partridge, 1987:2):

$$RTO_i = \beta \times SRTT_i \tag{1}$$

where

RTO_i = Retransmission Timeout

β = constant between 0 and 1 that is selected to mitigate the chances of the packet RTT exceeding the value of RTO_i

$SRTT_i$ = smoothed round trip estimate of average round trip times of sampled packets

In addition, $SRTT_i$ can be calculated as (Karn and Partridge, 1987:2):

$$SRTT_{i+1} = (\alpha \times SRTT_i) + (1 - \alpha) \times s_i \tag{2}$$

where

$SRTT_{i+1}$ = new value for the RTT estimate

α = constant between 0 and 1 that determines the speed to which SRTT adapts network changes

s_i = RTT samples

In the case of a transmission timeout (in other words, an ACK has not been received in the expected timeframe), the above equations are not used to estimate either RTO or RTT (as no new sample is available for use in estimation) and instead an arbitrary factor is used to increase RTO (some implementations double the previous value of RTO, others use a table of values to steadily increase RTO in the event of subsequent timeouts) (Karn and Partridge, 1987:3). This method of increasing RTO due to timeouts is known as back-off and this technique will continued to be used for RTO calculation until packet losses cease in occurrence (Karn and Partridge, 1987:3). Once the network is stabilized, RTO values will revert to estimates determined by equations (1) and (2). Finally, a sender will also retransmit a segment if a duplicate ACK is received for a given segment (Broyles, 1999:7).

TCP is also known as a sliding window protocol, which is a property that allows a sender to transmit a finite amount of segments before receiving an ACK from the receiver (Miller, 1998:2). In essence, the TCP window represents the amount of unacknowledged segments that can be in flight in a network at a given time. The sliding piece of the protocol is activated upon receipt of ACKs; receipt of ACKs allows the window to slide, meaning that additional segments can now be transmitted (Allman, 1997: 3). This property allows incremental growth in the number of segments allowed to transit a given network. Without this type of control, the amount of segments injected into a network could overwhelm available bandwidth allocations and result in the congestive collapse of the network (Floyd and Fall, 1999:459).

The maximum window size TCP allows a network device to advertise is 64 kilobytes (Allman, 1997:3; Allman, Hayes, Kruse, and Ostermann, 1997:3). Window sizes are negotiated between a sender and receiver during synchronization of a TCP session and are typically sized to allow an appropriate level of segment flow that will not cause network congestion or will allow flows that could exceed network bandwidth allocations. In addition to window strategies, TCP also utilizes four algorithms to mitigate network congestion that include Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery (Allman, Paxson, and Stevens, 1999:1). These algorithms are used to detect network congestion and to reduce the transmission rate of a network device to a level that can be supported by the available network resources.

TCP Congestion Control Algorithms

Congestion control algorithms utilize two state variables, Congestion Window (CWND) and Slow Start Threshold (SSTHRESH) (Allman, et al., 1999:8). CWND represents the amount of segments a device can inject into a network before receiving an ACK (Allman, Paxson, and Stevens, 1999:2). CWND will also be limited to the size of the advertised window of a receiver (Allman, et al., 1999:8). The CWND value can be increased or decreased based on the perceived amount of network congestion. SSTHRESH is used to determine what algorithm will be used to increase CWND (Allman, et al., 1999:8). If CWND is less than SSTHRESH, the Slow Start algorithm is used (Allman, et al., 1999:8). If CWND is greater than or equal to SSTHRESH, the congestion avoidance algorithm is used (Allman, et al., 1999:8). The initial SSTHRESH value will typically be the receiver advertised window size (Allman, Paxson and Stevens, 1999:4). The SSTHRESH value can also be set at the level in which network congestion was detected (Allman, et. al., 1999:9). While these algorithms are useful in preventing the congestive failure of a network, they also have a negative impact on the performance of TCP over a large delay or high bit-error rate network commonly encountered in a satellite-based transmission environment (Allman, 1997:3, Henderson and Katz, 1999:326).

Slow Start is the mechanism TCP utilizes to establish a network connection or restart a connection after a RTO has occurred (Allman, et al., 1999:9). The purpose of Slow Start is to ensure a network device does not transmit too large of a burst of data segments that could overwhelm the available network resources (Allman, et al., 1999:9). The Slow Start algorithm begins by initially setting the value for CWND to one segment and SSTHRESH to the receiver's advertised window size (Allman, et al., 1999:9). This limits the network device to the transmission of one segment and to wait for an ACK of receipt of that segment (Allman, et al., 1999:9; Carroll, 2004:7). For each ACK the transmitting device receives, the CWND is increased by one segment (Allman, et al., 1999:9). For example, after receipt of the initial ACK, the transmitting device will be able to send two segments (CWND equals two). After receipt of an ACK for each of the two segments, the sender will be able to send four segments (CWND equals four). This behavior represents an exponential growth pattern and will continue until CWND equals or exceeds the value of SSTHRESH or a segment loss is detected (Carroll, 2004:7). It is important to note that if the RTO expires for a transmitted segment, TCP will initiate the retransmission of the segment and will perceive the RTO expiration as a sign of network congestion (Allman, et al., 1999:10; Broyles, 1999:9). As a response to the perceived network congestion, TCP will reduce the transmission rate of a device by cutting the SSTHRESH value to half of the current CWND value and reset the CWND value to one, starting the Slow Start mechanism anew (Allman, 1997:4). When CWND equals or exceeds SSTHRESH, however, the Congestion Avoidance algorithm is used to further increase the size of CWND (Allman, et al., 1999:9).

The Congestion Avoidance algorithm is a more conservative measure to increase CWND than Slow Start and is primarily used to slowly probe the network for additional capacity (Allman, et al., 1999:9; Hoe, 1996:2). Congestion Avoidance only allows an increase in value of CWND if all segments transmitted in a window have a corresponding ACK, making the growth rate of CWND linear in nature (Carroll, 2004:7; Hoe, 1996:2). Mathematically, the congestion window will be increased by 1/CWND for each segment that is ACKed during use of the Congestion Avoidance algorithm (Carroll, 2004:7; Padhye, Firoiu, Towsley, and Kurose, 1998:3). From this property, CWND will be increased by roughly one segment for every round-trip time (Allman, 1997:5).

The Fast Retransmit and Fast Recovery Algorithms work in tandem to mitigate the time it takes a TCP session to return to the maximum transmission level of segments when packet loss or congestion is detected through the receipt of duplicate ACKs (Allman, et al., 1999:11). Under normal circumstances during a TCP connection, segments are assumed lost and are retransmitted when RTO occurs. Unfortunately, needless retransmissions of segments can occur despite successful transmission of a segment because the corresponding ACK is still traveling through the network or the segment waits for processing in a receiver’s buffer when RTO expires (Broyles, 1999:10). To counteract this, Fast Retransmit provides a way to retransmit a packet prior to RTO occurring by assuming that three duplicate ACKs correspond with a lost segment (Allman, 1997:5; Padhye, Firoiu, Towsley, and Kurose, 1998:3). In a simplified example in Figure 2, it can be seen that three identical ACKs cause the sender to retransmit segment three (keep in mind, the behavior outlined in Figure 2 does not exactly mimic a TCP session, but instead is for clarity in explaining Fast Retransmit).

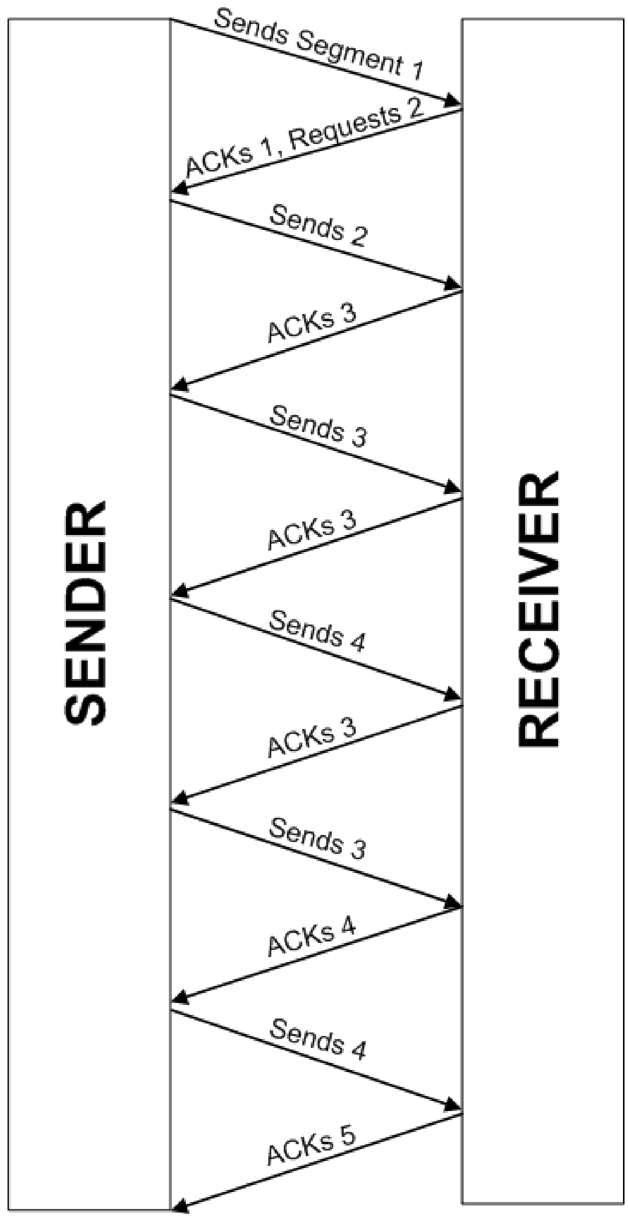

Figure 2. Fast Retransmit Example

When a segment is retransmitted by the Fast Retransmit protocol, the Fast Recovery Algorithm is activated in response to a perceived congestion of the network. The Fast Recovery Algorithm reduces the CWND to half the current value and resets the SSTHRESH value to the new value of CWND (Allman, 1997:6; Krishnan et al., 2004:344). The CWND is then artificially increased to match the number of duplicate

ACKs, under the assumption that duplicate ACKs indicate a lost segment that is no longer in the network and therefore additional network capacity exists (Broyles, 1999:10). Depending on the size of CWND, additional segments may be transmitted, however, the receipt of a non-duplicate ACK will reduce CWND to the value of SSTHRESH and will cause the start of the Congestion Avoidance algorithm (Allman, 1997:6).

Satellite Environment

The types of orbits utilized by constellations of satellite systems orbiting the earth include low earth orbit (LEO), highly elliptical orbit (HEO), and geosynchronous orbit (GEO). GEO satellites, particularly those that are part of the Defense Satellite Communications Systems constellations, are the most commonly utilized communication assets for deployed US military forces. Due to this fact, GEO satellites represent the transmission medium of interest for this paper. GEO satellites orbit at an elevation of approximately 36,000 kilometers above the surface of the earth (Roddy, 2001:14). At this elevation over the equator, satellites are able to achieve a speed that matches the rotational velocity of earth. This condition enables GEO satellites to remain stationary relative to a location on the equator, known as a subsatellite point. Due to a GEO satellite's high elevation and relatively stationary position, it is able to attain a coverage area of approximately ±75° latitude. Three geostationary satellites (spaced apart by 120° longitude along the equator) could provide whole earth coverage. For these reasons, GEO satellites have become a high demand asset during US military operations in areas with limited or no terrestrial telecommunications infrastructure, such as the Middle East.

As a result of a GEO satellite's high elevation, lengthy propagation times occur for radio signals traveling the slant distance between an earth station and the satellite. Propagation delays can vary from 239.6 to 279 milliseconds (ms) for one "hop" (ground station to satellite to ground station), depending on the location of ground stations in a satellite's spot beam (Allman, et. al., 1999:2). From these figures it can be calculated that the Round Trip Time (RTT) for a message and a corresponding response between two ground stations communicating via a GEO satellite could take between 479.2 and 588 ms. The propagation delay times could be even greater, depending on the time required for signal processing by the satellite and satellite motion characteristics (such as those space assets that have a "figure eight" motion near the end of operational life) (Allman, et al., 1999:2; Henderson and Katz, 1999:327). It will be shown later that this characteristic negatively impacts the throughput capability of a TCP-based connection.

TCP Limitations in a Satellite Environment

TCP's limitations in a satellite environment can be generalized into two areas: algorithms used to increase the sliding window size and TCP throughput in a high delay environment. The Slow Start and Congestion Avoidance algorithms, as outlined previously, determine the rate of size increase or decrease of the sliding window. The steady-state behavior of TCP determines the maximum throughput capability based on factors of window size and RTT. The time the Slow Start algorithm takes to reach a window size of W segments on a network with a given RTT (denoted by R) can be calculated as (Allman, 1997:7):

$$SS_{time} = R\log_2(W) \qquad (3)$$

Assuming a GEO satellite link with a RTT of 570 ms, 256 byte segments, and a maximum advertised window size (64 Kilobytes that would result in 256 segments) it would take 4.56 seconds to increase CWND to the advertised window size. In contrast, a terrestrial link with the same characteristics, but a RTT of 70 ms, would take 560 ms. These scenarios illustrate the potential waste in available bandwidth by TCP in a high delay environment when viewed in contrast to a low latency transmission medium.

As mentioned previously, Congestion Avoidance is utilized by TCP to slowly probe a network for additional capacity. The linear nature in which Congestion Avoidance increases CWND takes an exorbitant amount of time in a high loss or large delay transmission system. For example, if a loss (such as the loss of a segment) occurs on a satellite-based transmission network, the value of CWND is reduced to half of the original value. If 256 byte segments and a maximum window size were in use prior to the loss, the resulting new value for CWND would be 32 Kilobytes. Under these conditions (assuming a RTT of 570 ms), Congestion Avoidance would take 72.96 seconds to return CWND to the maximum window size. In contrast, a terrestrial link with similar properties, but a RTT of 80 ms, would take only 10.24 seconds to reach the maximum window size under Congestion Avoidance. This discrepancy in recovery time again highlights the potential waste of bandwidth utilizing TCP over high delay paths.

TCP also experiences a bottleneck in throughput capability in high delay transmission systems. Assuming a loss- and congestion-free network, the maximum throughput of TCP can be calculated as (Allman, 1997:10):

$$MT = \text{Receive Window Size}/RTT \tag{4}$$

With a maximum window size of 64 Kilobytes and a satellite link with a RTT of 570 ms, the maximum throughput of a TCP connection would approximately be 114,977 bytes per second. If the satellite link operated at a T1 (1.536 Mbits/second with removal of 8 Mbits for overhead) rate on a transponder, the link would only be 60% utilized (114,977 bytes/192,000 bytes). TCP would again fail to fully utilize the available bandwidth of the satellite transmission path.

Bit Error Rate (BER) Effects on TCP

When relying upon satellite transmission systems for communications, it is understood that BER will be significantly higher than those experienced with terrestrial transmission systems due to the atmospheric conditions that radio frequency signals must propagate through (Allman, et al., 1999:4; Roddy, 2001:444). Errors that occur due to this paradigm will cause the execution of the congestion control protocols of TCP, as TCP assumes any packet loss is due to network congestion (Abdelmoumen, Malli, and Barakat, 2004:3994; Caceres and Iftode, 1995:858; Krishnan et al., 2004:343; Ghani and Dixit, 1999:64; Miller, 1998:1). As a result, growth of CWND will be limited and will affect the throughput of a TCP session (Allman, et al., 1999:4; Narasimhan, Kruse, Ostermann, and Allman, 2004:1; Ghani and Dixit, 1999:64; Miller, 1998:1).

TCP Performance Enhancement Techniques

There are multiple strategies to enhance the performance of TCP over satellite links. The window scale option is one strategy in which a modification is added to the TCP header. During the initial synchronization of a TCP session, “SYN” segments are transmitted between sender and receiver. The SYN segment purpose is to establish the initial parameters (window sizes, segment lengths, etc.) for a TCP session. Adding the window scale option to the SYN segment would allow a sender to ascertain the available buffer size for a receiver. Based on the buffer size available and "agreement" between sender and receiver, a scale factor can be added to the TCP session parameters that could increase the maximum window size up to twice the value found in a standard TCP session (Jacobson and Braden, 1988:2; Mathis, et al., 1996:8). This relatively simple solution would allow increased bandwidth utilization for high delay paths based on the maximum throughput equation discussed previously.

Another option to improve TCP performance would be the use of selective acknowledgements (SACKs). SACKs allow receivers in a TCP session to communicate the success of transmission of every received segment (Jacobson and Braden, 1988:2). By bypassing the cumulative ACK system of standard TCP, senders would no longer need to wait for multiple RTT durations to determine what segments have been lost and will instead be able to retransmit specific lost segments in one or two RTT cycles (Ghani and Dixit, 1999:67-68; Mathis, et al., 1996:2). The SACK strategy mitigates the occurrence of RTO and results in the avoidance of activating TCP congestion control mechanisms that hinder throughput.

Performance Enhancing Proxies (PEP) Overview

PEPs can exist and act at any protocol layer(s), however, PEPs that are used to combat TCP degradation due to link errors or excessive latency are typically implemented at the application and transport layers (Border, et al., 2001:4). For the purposes of this research, the focus will be on PEPs utilized at the transport layer, as they are commonly utilized to enhance TCP performance (Border, et al., 2001:5). Transport layer PEPs make use of multiple methods to influence TCP behavior. These methods include modifying ACK spacing (to prevent their accumulation and subsequent "bunching" that can inflate RTO occurrences) and the generation/transmission of ACKs to local devices in order to minimize the time need to reach the full window size and, as a result, maximize throughput (Border, et al., 2001:5). It should be mentioned that not all PEPs utilize a standard set of algorithms/protocols to manipulate TCP sessions. Some PEPs, such as those produced by Mentat Inc., have made use of proprietary protocols to improve TCP performance. Other manufacturers have opted to use an open standard protocol known as SCPS-TP. Regardless of the underlying protocols, PEPs will typically be geared towards improving TCP through manipulation of the window size and ACK schema (in essence minimizing or eliminating the occurrence of the congestion control algorithms) through the modification of on-going TCP session or by splitting the TCP session (Ehsan, Liu, and Ragland, 2003: 514). TCP session splitting means that the PEPs on either side of the link will negotiate TCP sessions with local area network entities; TCP actions, such as the ACK system, will be handled by the PEP on behalf of the sender/receiver (depending on which side of the link the PEP resides) (Ehsan, Liu, and Ragland, 2003: 514). This allows a session's window to grow more quickly, as the ACK response is now handled locally and does not require ACK transmission over the satellite path (Ehsan, Liu, Ragland, 2003: 514). PEPs that split TCP sessions will typically provide segment (and requisite ACK results) information to each other in order to monitor and predict network loss/congestion and to allow the appropriate TCP response to network degradations.

SCPS-TP Overview

SCPS-TP was a joint development effort between the MITRE Corporation, MCI, NASA, and the DoD (Durst, Miller, and Travis, 1997:389). As with most TCP improvement efforts, SCPS-TP was created to overcome limitations associated with high delay and error environments; of particular interest were those transmission systems that made use of space-based assets and suffered from such environmental limitations. To improve TCP performance, SCPS-TP makes use of a number of techniques: header compression, Selective Negative Acknowledgements (SNACKs), TCP Timestamps, window scale modifications, and the TCP Vegas congestion control algorithms (Durst, Miller, and Travis, 1997:391-396).

Header Compression

The SCPS-TP standard seeks to maximize the amount/use of available bandwidth in a given transmission system. One way SCPS-TP accomplishes this is through use of a header compression schema. Header compression, under SCPS-TP, lessens the amount of header information passed between a sender and a receiver during a TCP session (Durst, Miller, and Travis, 1997:394). This is accomplished by only allowing changed header information to be passed directly during the TCP session; static header information is summarized and omitted fields (those TCP header fields whose flags are not set) are not passed (Durst, Miller, and Travis, 1997:394; Ishac, 2001: 6-7). Though this compression technique can result in variable lengths for the TCP header, it can result in up to a 50% decrease in header size (Durst, Miller, and Travis, 1997:394).

Selective Negative Acknowledgements

The primary purpose of the SNACK option is to improve the loss recovery mechanisms of a TCP session, while at the same time improve the bandwidth utilization between a sender and receiver (Durst, Miller, and Travis, 1997:394). This goal is met by using some of the properties of the SACK option, presented in RFC 2018 (Mathis et al., 1996:1-12), and the Negative Acknowledgement (NAK) proposal found in RFC 1106 (Fox, 1989:1-13; Durst, Miller, and Travis, 1997:394-395). Unlike the SACK and NAK options (and the ACK system found in generic TCP), the SNACK solution can enable a receiver to identify multiple holes (missing segments) to a sender (Broyles, 1999:15; Durst, Miller, and Travis, 1997:395; Luglio, Cesare, and Gerla, 2004:4). This property allows more efficient use of transmission bandwidth, as the SNACK option needs just a single RTT allocation to identify multiple missing segments, whereas the SACK and NAK options require a RTT allocation for each missing segment notification (Broyles, 1999:15).

Timestamps and Window Scale Modifications

The SCPS-TP modification makes use of the TCP Timestamp and Window Scale Modification (Window Scale Modification found in SCPS-TP are the same as those previously discussed in the TCP Performance Enhancement Techniques found in this section and therefore will not be covered in this part of the document) options found in RFC 1323 (Jacobson, Braden, and Borman, 1992:1-37; Durst, Miller, and Travis, 1997:396). The TCP timestamp option enables a receiver to add timing information in each ACK sent to a sender during a TCP session (Jacobson, Braden, and Borman, 1992:12-16). This allows the sender to approximate the RTT required for segment/ACK transmission; this knowledge of the transmission medium by the sender allows it to determine whether or not “missing” segments need to be transmitted, as these segments may still be in transit or in the receiver’s buffer based on the calculated RTT and the segment size (Durst, Miller, and Travis, 1997:396; Jacobson, Braden, and Borman, 1992:14).

TCP Vegas Congestion Control Algorithms

The TCP Vegas variant was developed to further increase the transmission throughput of a network, as well as minimize the network’s segment loss when compared to the generic TCP standard and other associated variants (such as TCP Reno and TCP Tahoe) (Brakmo and Peterson, 1995: 1468). To accomplish this goal, the creators of TCP Vegas made modifications to the TCP standard’s algorithms concerning segment retransmission response time (in the event of segment loss, the RTT calculation is more accurate and belies the need for three duplicate ACKs in order for retransmission of a segment to occur), made possible for the TCP mechanism to foresee network congestion and change transmission rates in response (TCP Vegas compares current throughput rates with the expected throughput rate based on network bandwidth allocations to expand the TCP window; generic TCP and TCP Reno continually grow the TCP window until a loss occurs that could potentially invoke other congestion control algorithms), and evolved the TCP slow-start algorithm to limit loss during its execution (TCP Vegas allows only exponential growth of the window, after a loss, every other RTT (linear growth allowed on the other cycles) in order to minimize the potential of allowing the window size to exceed network capabilities, but still allow the window to reach maximum size in a timely fashion (Brakmo and Peterson, 1995: 1468-1480).

Summary

This chapter presented an overview of the TCP/IP protocol suite, to include the congestion control algorithms. TCP/IP performance hinderers, to include the satellite environment (latency), BER, and bandwidth asymmetry effects, were also presented. Improvements to the TCP algorithms, namely SACKs and an increased widow size, to counteract the above effects were covered, as were PEPs (particularly those that make use of the SCPS-TP mechanisms) that are also utilized to improve TCP performance due to degradations.

III. Methodology

Chapter Overview

This chapter will present an overview of the methodology to the experimentation that is required for this study. Items to be covered include the justification for the experimental method, and an overview of the test environment and associated hardware and software. In addition, the experimental design, to include identification of dependent and independent variables, will be presented, as well the statistical methods that will be used to examine obtained data sets.

Method Selection

In studying network topologies there are generally three methods of experimentation: direct study, mathematical derivation, and simulation (Broyles, 1999:19-20). Direct study involves obtaining experimental measures from actual networks of interest; in the case of this study, this means access to a satellite transmission system, as well as associated TCP/IP networks that utilize the satellite transmission path. Due to the relatively high demand and low availability of such assets, this type of experimentation was not considered a viable solution. Mathematical derivation of network performance would typically involve solving a copious amount of simultaneous equations (Broyles, 1999:20). This type of experimentation was considered beyond the scope of this body of work. The final method involves the creation of a simulated environment in which to test various independent variables effect upon dependent variables of interest. This type of method allows the experimenter a great amount of control concerning the application of independent variables, as well as measuring the response of dependent variables. In addition, the virtual creation of infrastructure objects (such as a satellite terminal) gives access to assets that would other wise be unavailable. For these reasons, simulation was the method of choice for the experimentation required for this thesis work.

Simulation Environment

The simulation environment of this study was not created via a network simulation tool, such as OPNET, but instead was constructed through the implementation of a computer network and use of software packages to simulate various network/transmission devices. See the sections below for an overview of the network topologies and software utilized for this thesis work.

Network Topology for TCP/SACK/Window Scale Testing.

The computer network implemented for this research consisted of five Dell 1750 Power Edge blade servers connected via gigabit Ethernet network interface cards (NICs) and category five crossover cables. For a listing of hardware specifications for these servers, refer to Table 1. For the network topology, see Figure 3.

Table 1. Server Hardware Listing

Manufacturer	Dell
Model Number	1750 Power Edge
Processor Type	Intel Xeon 2.4GHz, 533MHz FSB
Processor Configuration	Dual
Memory	1024Mb ECC DDR
Hard Drive Capacity	2 x 36GB
NIC Type	Gigabit Ethernet
NIC Manufacturer	Broadcom

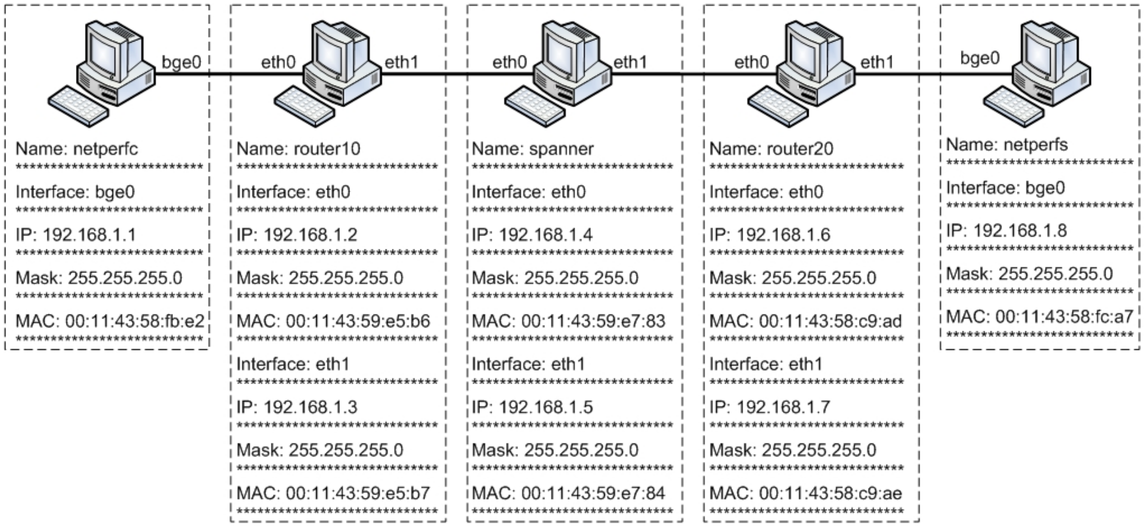

Figure 3. Network Topology

The first step taken to establish the test network was the assignment of IP addresses to each of the respective servers and associated Ethernet interfaces. As can be seen in Figure 3, the three middle servers have a dual NIC configuration that necessitated the assignment of two IP addresses. The end machines (*netperfc* and *netperfs*) had gateway links established to their neighboring servers (*router10* and *router20*); the remaining servers had IP forwarding enabled (allows the respective machines to pass through IP traffic) and had networking routing information for each of the IP networks established for each Ethernet interface via the manipulation of network system files. End-to-end connectivity was verified via successful use of ping and traceroute commands on each of the servers. Also noteworthy is the use of a single network for use in this experimentation. The program that was used for satellite simulation, spanner, implements a bridging function that necessitated the use of a single network between *router10* (eth1 interface), *spanner* (eth0 and eth1 interfaces), and *router20* (eth0 interface). In addition, PEPs were utilized to test the SCPS enhancement; these PEPs were also implemented via a bridging function. To simplify routing setup, it was decided that all servers would reside on the same IP network.

The servers depicted in Figure 3 were loaded with an open source operating system (OS); Fedora Core 4 was loaded on the edge servers (*netperfc* and *netperfs*) and the middle server (*spanner*). Fedora Core 2 was installed on boxes designated *router10* and *router20*. The OS selection was based upon the native environment required for software packages used to emulate various network devices. Iperf, a software tool that is used to gather network performance metrics, operates natively in a Linux-based environment with Kernel builds 2.1 or greater. Spanner, a satellite link emulator developed by the Mitre Corporation, is also native to a Linux environment with successful loads occurring on boxes utilizing Kernel build 2.4 or greater. Iperf was loaded on hosts netperfc and netperfs, whereas spanner was installed on host spanner. A brief overview of both Iperf and spanner can be found in the following sections. The *router10* and *router20* systems did not have any software loaded on them other than the Fedora Core 2 OS; a Linux-based OS supports various TCP enhancement protocols, most notably TCP extensions for window scale modifications and SACKs as conceived by RFC 1353. The OS implements TCP extensions by default; both window scale and SACK techniques can be individually enabled or disabled through manipulation of network system files core to the Linux OS.

Iperf Overview.

Iperf is a freely distributed software package that can be utilized to measure a number of network performance metrics, to include network throughput (Blum, 2003:99-101). Iperf obtains network information through use of a client/server topology. A networked computer will execute Iperf and in essence represent the client side of the program. A separate machine will also implement Iperf, but will instead act as a server. An Iperf server awaits inbound connections from remote hosts that are executing Iperf commands from the client side (Blum, 2003:105). From a command line interface (Iperf also includes a package known as Jperf that is a Java-based graphical interface; Jperf was not used during this experimentation) Iperf can pass specified traffic patterns between the client and server and generate subsequent network performance metrics (Blum, 2003:101-102). Such traffic patterns can include TCP stream testing in which maximum throughput (by default Mbits/sec, can be modified) of a TCP session can be determined (Blum, 2003:101). The command line interface also allows manipulation of various test factors, such as packet size and test duration, allowing traffic patterns to be tailored in order to measure specific network loading situations (Blum, 2003:106-107).

Spanner Overview.

The Mitre Corporation developed the spanner software package to provide satellite link emulation for protocol testing environments composed of networked computers (Mitre, 2006). Spanner is typically loaded on a system that is used to bridge two or more computers that are acting as network devices, such as client, servers, or routers. In this way, a machine that is executing spanner is acting as the transmission path for other networked devices. Conveniently, spanner allows manipulation of key satellite propagation characteristics via a command line interface; variables that can be emulated by spanner include latency, BER, and bandwidth allocation (Mitre, 2006). Through manipulation of these variables, a realistic transmission environment can be virtually constructed.

Network Topology for PEP/SCPS-TP Testing.

The experimentation for this study also necessitated another network topology in order to ascertain the performance benefits of a PEP that utilizes the SCPS-TP standard. As can bee seen in Figure 4, the network topology for this segment of testing replaces the servers used as routers with PEP devices from Xiphos Technologies, Inc.

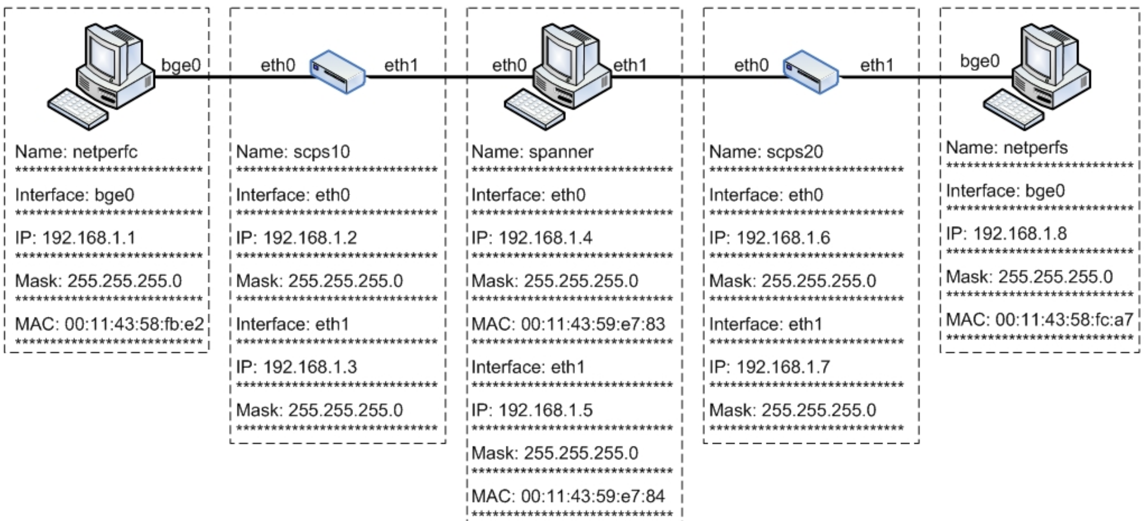

Figure 4. Network Topology for PEP Testing

All other servers remain unchanged and will continue to utilize onboard software to perform network metrics gathering. An overview of the XipLink Mini-Gateway© is presented in the following section.

Xiplink Mini-Gateway Overview.

The Xiplink Mini-Gateway makes use of the SCPS-TP standard and is intended for use in transmission environments that experience high levels of latency as well as excessive amounts of bit errors (Xiphos Technologies, Inc., 2005:1). The Mini-Gateways make use of two 10/100 Ethernet interfaces; one interface is intended for connection to a LAN, the other for connection to either a router or direct connection to a satellite modem (Xiphos Technologies, Inc., 2005:3-6, 31). In addition, the PEPs can operate in two modes, termed routing and bridging (Xiphos Technologies, Inc., 2005:3-4). Routing mode, based on the appropriate IP configurations, allows a near "drop and insert" installation in any network environment that requires TCP session enhancement. The bridging mode is intended for use on networks that pass traffic via satellite links (hence, needing enhancement) and traffic that does not travel on high delay/error paths (no enhancement needed) (Xiphos Technologies, Inc., 2005: 4.) Bridging mode also requires minimal routing information during setup of a Xiplink; only default routes need be specified, as well IP assignments to a given device's Ethernet interfaces. Routing mode setup can also be minimal, particularly if a Dynamic Host Configuration Protocol server is used. In the case of static routing, routing mode setup can be more cumbersome. In order to simplify setup, due to use of static routing, it was decided that bridging mode would be the preferred method. It is of importance to note that bridging mode in the Xiplink mini-gateway does not allow use of header compression specified in the SCPS standard. Use of header compression is, however, available in the routing mode implementation. Unfortunately, firmware errors concerning header compression have been identified by Xiphos Technologies making, in essence, header compression unavailable in either mode. It is expected that header compression will make litter difference in the magnitude of throughput testing results. Screenshots of the configuration of the respective Xiplinks can be seen in Appendix A.

Experimental Design

The overall design for experimentation will be a cross-section factorial design. This design type was selected as it minimizes the amount of sample points required and allows for direct analysis of interactions between variables of interest (Schwab, 2005:78). The dependent variable of interest will be average throughput (kilobits/sec) as calculated by Iperf during testing of the two network topologies outlined previously. It should be noted that the throughput output provided by Iperf is an average of samples obtained during the test duration. The independent variables of interest can generally be separated into two categories: TCP enhancement technique(s) and transmission factors. TCP enhancement techniques include use of standard TCP (no enhancements, acts as the control data), use of SACKs (on all servers), use of window scale (all servers), use of both SACKs and window scale techniques (all servers), and use of a PEP/SCPS-TP standard. See Table 2 for the Linux kernel settings for each of the testing scenarios.

Table 2. TCP Testing Server Settings

Test Scenario	Kernel Setting (Modification of /etc/sysctl.conf)	Servers Modified
TCP	None	None
Window Scale	net.ipv4.tcp_window_scaling = 1	All
SACK	net.ipv4.tcp_sack = 1	All
Window Scale/ SACK	net.ipv4.tcp_sack = 1 net.ipv4.tcp_window_scaling = 1	All
SCPS-TP	None	None

Note that all of the servers using a TCP enhancement(s) must be modified in order to ensure a given enhancement was used end-to-end. It was observed during dry run testing that modification (either window scale, SACK, or window scale/SACK) of only the client (*netperfc*) and server (*netperfs*) resulted in performance values that were comparable to standard TCP results. Conversely, modification of the entire topology resulted in improved throughput performance, as compared to standard TCP, for each of the TCP enhancement strategies. The likely reason for this is that unmodified servers were not passing the appropriate header flags to neighboring machines concerning use of SACKs; servers that did not make use of window scale were likely minimizing the agreed to window size during a given TCP session start-up. Modifying all servers in the transmission chain allowed each server to pass the appropriate header information and agree upon an appropriate window size and make use of SACKs as outlined in RFC 1353.

The transmission factors that will be used during testing include delay (fixed at 285ms for one-way propagation), BER (to include levels of 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, and 10^{-9}), transmission bandwidth (1.544Mb/s) and packet size (128kb, 512kb). Additionally, scenarios will be conducted in five-minute intervals. For a listing of independent/dependent variables used for each test scenario, refer to Tables 2, 3, 4, 5, and 6 below. Please note that for every scenario listed below, 30 samples (simulation time will be 5 minutes per sample) will be taken (150 minutes of simulation per scenario). In addition, each scenario was conducted in an automated fashion via scripting; see Appendix B for the syntax of the utilized scripts.

Table 3. TCP Testing Dependent/Independent Variables

Dependent Variable	Metric
Average Throughput	kb/s
Independent Variables	Treatment Levels
SACKs	No
Window Scale	No
SACKs/Window Scale	No
SCPS-TP	No
Delay	570ms, Round Trip
BER	10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}
Transmission Bandwidth	1.544Mb/s
Packet Size	128kb, 512kb
Duration	300 sec

Table 4. SACK Testing Dependent/Independent Variables

Dependent Variable	Metric
Average Throughput	kb/s
Independent Variables	Treatment Levels
SACKs	Yes
Window Scale	No
SACKs/Window Scale	No
SCPS-TP	No
Delay	570ms, Round Trip
BER	10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}
Transmission Bandwidth	1.544Mb/s
Packet Size	128kb, 512kb
Duration	300 sec

Table 5. Window Scale Testing Dependent/Independent Variables

Dependent Variable	Metric
Average Throughput	kb/s
Independent Variables	Treatment Levels
SACKs	No
Window Scale	Yes
SACKs/Window Scale	No
SCPS-TP	No
Delay	570ms, Round Trip
BER	10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}
Transmission Bandwidth	1.544Mb/s
Packet Size	128kb, 512kb
Duration	300 sec

Table 6. SACK/Window Scale Testing Dependent/Independent Variables

Dependent Variable	Metric
Average Throughput	kb/s
Independent Variables	Treatment Levels
SACKs	No
Window Scale	Yes
SACKs/Window Scale	Yes
SCPS-TP	No
Delay	570ms, Round Trip
BER	10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}
Transmission Bandwidth	1.544Mb/s
Packet Size	128kb, 512kb
Duration	300 sec

Table 7. SCPS-TP Testing Dependent/Independent Variables

Dependent Variable	Metric
Average Throughput	kb/s
Independent Variables	Treatment Levels
SACKs	No
Window Scale	No
SACKs/Window Scale	No
SCPS-TP	Yes
Delay	570ms, Round Trip
BER	10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}
Transmission Bandwidth	1.544Mb/s
Packet Size	128kb, 512kb
Duration	300 sec

Statistical Analysis Techniques

Upon successful gathering of the experimental data it will of course be necessary to analyze the data in a manner that will answer the research questions specified in Chapter 1. Most of the research questions deal with performance benefits of the various enhancement techniques as compared to a standard TCP implementation. These type of questions can generally be answered utilize descriptive statistics. The more probing question of the study deals with the determining whether or not a statistical difference exists between the performance improvements afforded by each of the TCP enhancement techniques. Answering this question will require use of more vigorous statistical analysis than descriptive statistics can provide. Therefore, it will be necessary to make use of a statistical software package, in the case of this study JMP 5.1, to generate the necessary statistical analyses. JMP 5.1 will be used to conduct an analysis of variance (AOV) model in which the null hypothesis will be that the means of the data being compared are equal, or (McClave, Benson, and Sincich, 2005:567):

$$H_o: \mu_1=\mu_2=\mu_3=\mu_4=\mu_5$$

The alternative or research hypothesis is that at least one of the means of the data being compared are not equal, or (McClave, Benson, and Sincich, 2005:567):

$$H_a\text{: at least one mean is different}$$

The AOV output under JMP 5.1 will provide an R^2 adjusted value and a p-value that will reveal the fitness of the model hypotheses. It will, of course, be necessary to validate the three key assumptions (normality, constant variance, and independence of the residuals) of the AOV method before any generalizations can be drawn from the results. In addition, as a result of conducting an AOV via JMP 5.1, another analysis tool known as Tukey analysis becomes available; Tukey analysis aids in determination to the statistical differences (if any) between sets of data. This will further aid in the statistical analysis of the gathered data sets.

Data analysis of experimental results does not always progress in an expected manner. In the case of the AOV method, there is a distinct possibility that data sets may not meet the three assumptions of normality, constant variance, and independence. Therefore it may be necessary to make use of a non-parametric technique that does not require any type of assumptions concerning the data distribution(s). One such technique, the Kruskal-Wallis H-test, allows comparison of means without any type of assumptions of the probability distributions (McClave, Benson, and Sincich, 2005:1095). Much in the same manner as AOV, the Kruskal-Wallis test allows for differences between populations to be detected. Unlike AOV, Kruskal-Wallis accomplishes this by ranking (from smallest to largest) obtained data points; rankings are conducted as though samples were obtained from the same population, meaning, in the case of this study, samples obtained for scenarios that utilize difference enhancement techniques would be grouped together during the ranking process (McClave, Benson, and Sincich, 2005:1079, 1096). The H-statistic generated by the Kruskal-Wallis test measures the differences between samples according to the generated rank structure (McClave, Benson, and Sincich, 2005:1097). In terms of a null or alternative hypothesis, the Kruskal-Wallis test can be expressed as (McClave, Benson, and Sincich, 2005:1098):

H_o: probability distributions are identical

H_a: at least two of the probability distributions differ in location

It is key to note that the JMP 5.1 software package does not express the H-statistic directly, but instead generates a ρ-value that can be compared to the model's error probability (JMP 5.1 defaults to an α value of .95, with the probability of an error at .05). A valid test will have a ρ-value, in the case of the default reliability of JMP 5.1, less than .05.

The statistical analyses of this research can aid military units and commercial entities in deciding and justifying use (or opting not to use) of a TCP enhancement strategy. Obviously, units will need to take into account the costs of their respective satellite leases and balance the gains (in terms of bandwidth efficiency) a TCP enhancement can provide versus its costs. This business case analysis will be discussed further in the following chapters.

Chapter Summary

This chapter presented an overview of the research methodology for this study. Items presented in this chapter included the justification for the method of experimentation, an overview of the test network topologies to include hardware and software considerations, the experimental design, and an overview of the statistical tools that will be used on the gathered data sets.

IV. Results and Analysis

Chapter Overview

This chapter will present the data analyses for this thesis work. Data analysis was conducted in sections, based on the level of BER applied and the size of packets transmitted from client *netperfc* and server *netperfs*. As a result, 10 sections of data analysis will be covered; summary information will follow these sections that present the findings of the performance of the enhancement techniques under consideration.

Statistical Analyses

The statistical analyses in this section were conducted in the manner outlined in Chapter 3. The data analysis presentation will be rather thorough for scenario 1; all other scenarios, unless otherwise specified, will follow along the same path as scenario 1 and as such, their respective findings will be summarized appropriately. The complete data set, for which these statistical analyses were conducted on, can be found in Appendix C.

Scenario 1: 10^{-5} BER and 128kb packets

The execution of an AOV fit model via JMP 5.1 for this scenario resulted in relatively high values for R^2 (.92) and R^2 (.92) adjusted; as expected, the values for these results were close in value (after rounding, identical), as R^2 was not inflated due to excessive model factors (all models only took throughput into consideration versus the enhancement technique utilized). In addition, the obtained ρ-value was less than .0001, indicating further validation to the model's fitness and that a difference exists between at least two of the sample means (H_a is found to be true). See Tables 8 and 9 for the analysis of variance and summary of fit results.

Table 8. Analysis of Variance Results for 10^{-5} BER/128kb Packet Scenario

Output	Result
R^2	.92
R^2 Adjusted	.92
Root Mean Square Error	12.68
Mean of Response	66.32
Observations	150

Table 9. Model Summary of Fit for 10^{-5} BER/128kb Packet Scenario

Output	Result
Degrees of Freedom, Model	4
F-Ratio	418.17
Probability > F (ρ-Value)	<.0001

The resulting Tukey analysis from this model established two groupings with respect to the enhancement technique utilized (at a family-wise error rate of .05). The SCPS-TP enhancement comprised one group, with all other techniques making up the other group. The Tukey analysis results can be seen in Table 10.

Table 10. Tukey Analysis Results for 10^{-5} BER/128kb Packet Scenario

Enhancement Technique	Grouping*		Least Square Mean
SCPS-TP	A		150.67
Window Scale/SACK		B	49.49
SACK		B	49.03
Window Scale		B	41.79
TCP (No Enhancement)		B	40.64

These results indicate definite performance differences, but before any inferences can be made, it will be necessary to validate the AOV model assumptions concerning independence, normality, and constant variance of the residuals. To test for independence, the Durbin-Watson test was utilized; for normality, the Shapiro-Wilk test; and finally, for constant variance the Breusch-Pagan test was used.

The Durbin-Watson test results indicated a lack of independence of the residuals, based on the obtained ρ-value of .0126 (at a passing threshold of .05). This violation has

*Groups not connected by the same letter are significantly different serious repercussions in a time series regression model (such as temperature forecasting), of which this study is not categorized as one (McClave, Benson, and Sincich, 2005:1044-1048). Based on the closeness in value of the obtained ρ-value and the threshold for a failed to reject result, it can be surmised that this violation is minor.

The Shapiro-Wilk test result also casts doubt on the validity of the AOV model, with the goodness-of-fit calculation resulting in a ρ-value that was less than .0000. Though the residual distribution in Figure 5 appears to be somewhat symmetric and normal in shape, there is a significant amount of outliers existing beyond three standard deviations from the majority of residuals indicated by the box-plot.

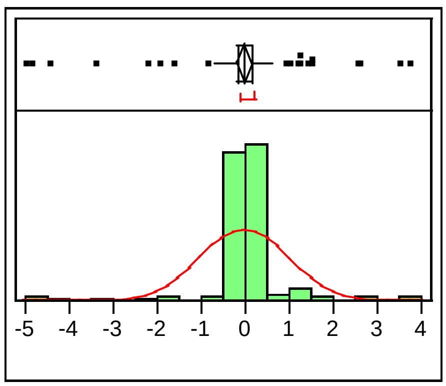

Figure 5. Distribution/Box-Plot for Throughput Residuals for 10^{-5} BER/128kb Packet Scenario

AOV is robust against violations of normality, but it is still of interest to determine why such a violation exists (White, 2005). In plotting the distribution for throughput (all results), it can be seen in Figure 6 that the sample data clusters around two distinct areas;

one exists below the value of 100, whereas the other exists above this reference point.

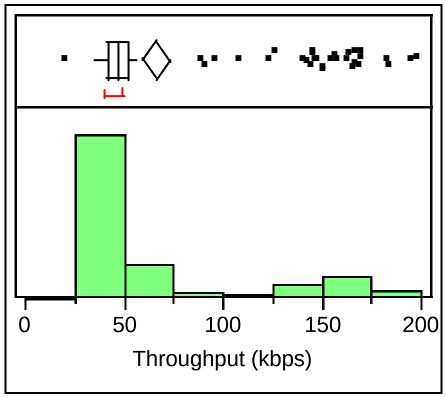

Figure 6. Throughput Distribution for 10^{-5} BER/128kb Packet Scenario

From review of the data set, these separate populations were created due to the clumping of data points around 50kbps (the throughput results of generic TCP, SACK, window scale, and window scale/SACK) and at 150kbps (SCPS-TP). The resulting skewness of the data set is likely responsible for the violation of normality of the residuals; though AOV is robust against this violation, it is of concern with respect to the validity of this model.

In order to execute Breusch-Pagan test, it is necessary to obtain the values for the Sum of Squares (SSR) from a fit model that uses the square of the residuals (residuals^2) and from the Sum of Squares Error (SSE) from the original (fit model of throughput) model. Conducting a fit model with residuals^2 resulted in an SSR of 12,883,090; the original model had a SSE of 23,297.73. The ρ-value for the Breusch-Pagan test was obtained via use of a spreadsheet program (Microsoft's Excel) and the expression:

CHIDIST (((SSR/2)/(SSE/N)^2),DF) (5)

Where N is the number of observations and DF represents the model's degrees of freedom. In actual execution, the spreadsheet expression refers to cells of the spreadsheet; these references were changed to variable representations for clarity. Calculation of the ρ-value for this test resulted in a value less than .0000. At a threshold of .05, the model has failed to pass this test, indicating a large amount of variance exists in the model. Due to the results of this test and the other validation tests for AOV, it is apparent that inference-making capability afforded by the model is extremely limited. As a result, it will be necessary to conduct a non-parametric test, the Kruskal-Wallis rank sums test, which requires no assumptions concerning the sample distribution and provides a tool to gauge whether or not a difference exists between the differing enhancement techniques.

The Kruskal-Wallis test revealed, through observation of the rank sums score means and the obtainment of a ρ-value of less than .0001, that there is a statistical difference in the probability distributions for each of the enhancement techniques, meaning that the research hypothesis, H_a, has been verified. The results of this test can be seen in Table 11 below.

Table 11. Kruskal-Wallis Test Results for 10^{-5} BER/128kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	2642.5	88.08
SCPS-TP	30	4065	135.5
TCP (No Enhancement)	30	858.5	28.62
Window Scale	30	1002.5	33.42
Window Scale/SACK	30	2756.5	91.88

Unfortunately, though the Kruskal-Wallis method can determine differences between samples, it is unable to adequately group these samples based on statistical equivalence as found in the Tukey analysis. Since a difference has been statistically determined, however, one can now rely upon descriptive statistical techniques to determine the differences between the TCP enhancement techniques.

The first descriptive method used will be the determination of confidence intervals (at a reliability of .95) for each of the enhancement techniques with respect to measured throughput. Using Excel's data analysis add-in, confidence intervals can be quickly generated; the results can be seen in Table 12.

Table 12. Descriptive Statistics for 10^{-5} BER/128kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	49.03	2.85	8.13	49.03 ± 1.06
SCPS-TP	30	150.67	27.69	766.94	150.67 ± 10.3
TCP (Unmodified)	30	40.64	4.21	17.7	40.64 ± 1.57
Window Scale	30	41.78	2.08	4.34	41.7867 ± 0.77
Window Scale/SACK	30	49.49	2.48	6.18	49.49 ± 0.92

From Table 12, the confidence intervals reveal groupings of the enhancement techniques. SCPS-TP can be placed in its own group, as no other method comes close to its performance numbers. The SACK and window scale/SACK methods can be grouped together, as they appear to provide the same level of benefit concerning throughput, which is marginally better than generic TCP and the window scale strategy. Finally, the window scale technique seems to be equivalent to the baseline measure of unmodified TCP. In terms of throughput enhancement, Figure 7 displays the percentage improvement afforded by each of the techniques as compared to the baseline (unmodified TCP); Figure 8 demonstrates the amount of bandwidth utilization for each of the modifications, to include standard TCP.

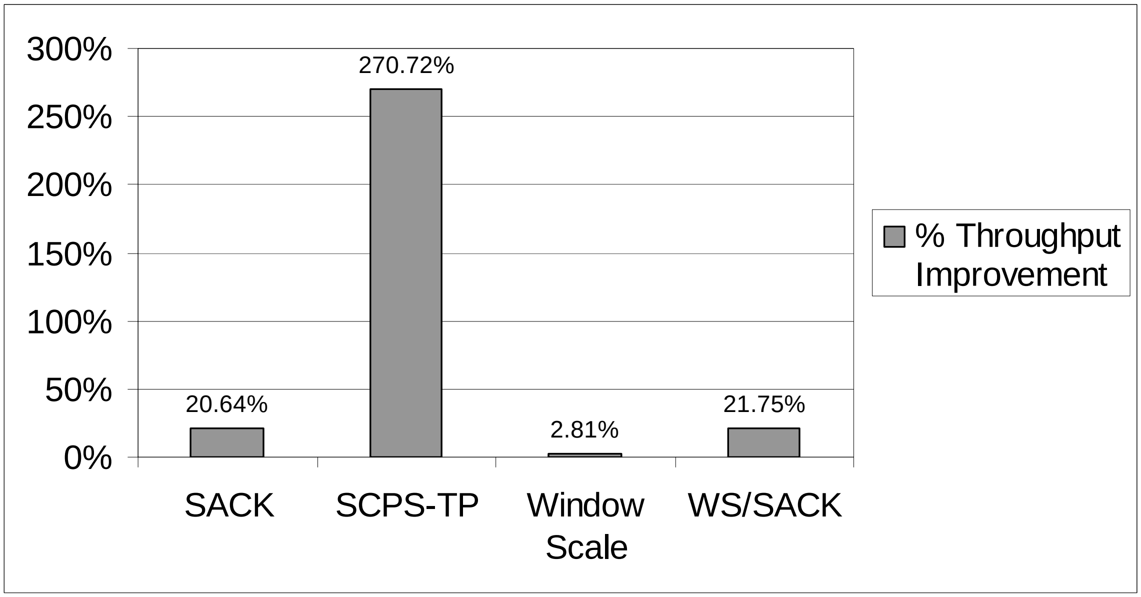

Figure 7. Throughput Improvement Percentages for 10^{-5} BER/128kb Packet Scenario

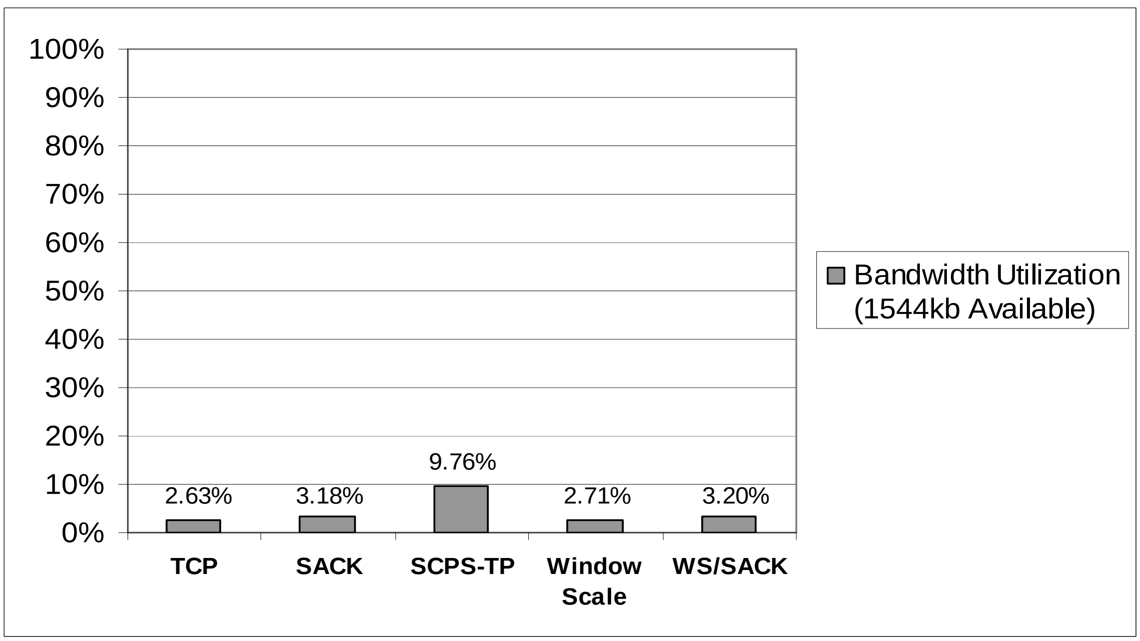

Figure 8. Bandwidth Utilization for 10^{-5} BER/128kb Packet Scenario

The remaining scenarios where analyzed in the same fashion as Scenario 1 and were also found to violate the required AOV assumptions, particularly that of constant variance. As a result, the remaining scenarios will not present the results of the AOV analysis and will instead present just the non-parametric results.

Scenario 2: 10^{-5} BER and 512kb packets

The Kruskal-Wallis test revealed, with a ρ-value less than .0001, a statistical difference in the probability distributions for each of the enhancement techniques exists. The results of this test can be seen in Table 13 below.

Table 13. Kruskal-Wallis Test Results for 10^{-5} BER/512kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	2723	90.77
SCPS-TP	30	4065	135.5
TCP (No Enhancement)	30	802.5	26.75
Window Scale	30	1041.5	34.72
Window Scale/SACK	30	2693	89.77

Knowing that a statistical difference exists between at least one of the enhancement methods, descriptive statistics can now be used to give insight as to what those differences are. A 95% confidence interval was calculated for each of enhancement methods with respect to throughput. The results of these calculations can be seen in Table 14.

Table 14. Descriptive Statistics for 10^{-5} BER/512kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	49.03	2.11	4.47	49.03 ± 0.79
SCPS-TP	30	150.34	20.57	424.07	150.34 ± 7.68
TCP (Unmodified)	30	41.06	1.81	3.28	41.06 ± 0.68
Window Scale	30	42.01	1.9	3.62	42.01 ± 0.71
Window Scale/SACK	30	49.02	2.75	7.55	49.02 ± 1.03

From Table 14, the confidence intervals reveal groupings of the enhancement techniques. SCPS-TP can be placed in its own group, as no other method comes close to its performance numbers. The SACK and window scale/SACK methods can be grouped together, as they appear to provide the same level of benefit concerning throughput, which again is marginally better than generic TCP and the window scale strategy.

Finally, the window scale technique seems to be equivalent to the TCP baseline. The percentage of throughput enhancement, with respect to TCP, can be seen in Figure 11.

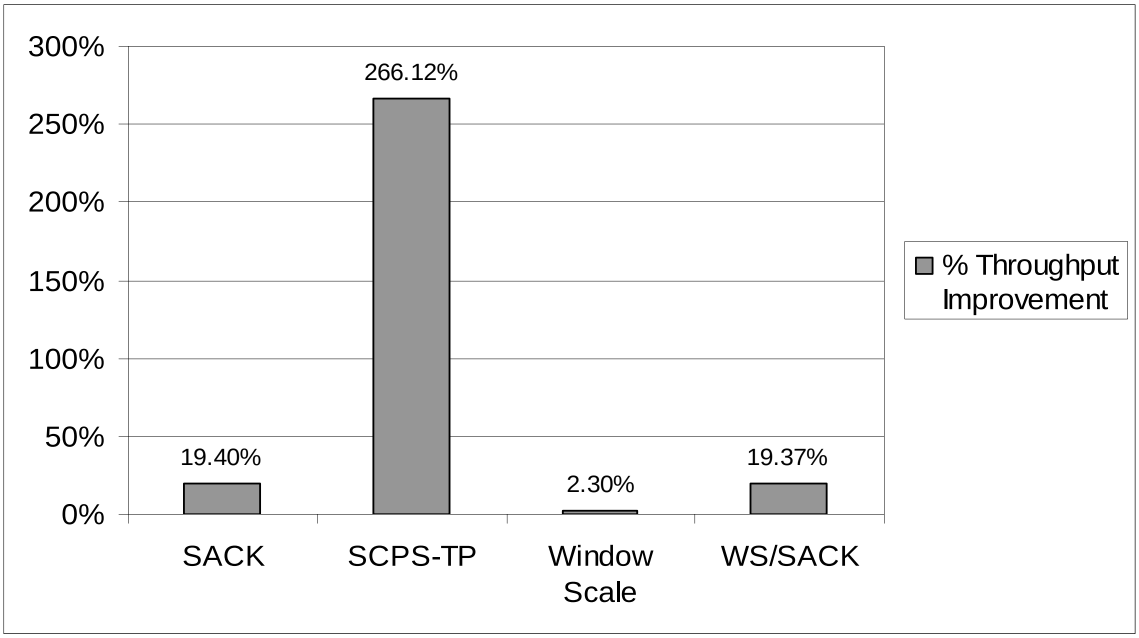

Figure 9. Throughput Improvement Percentages for 10-5 BER/512kb Packet Scenario

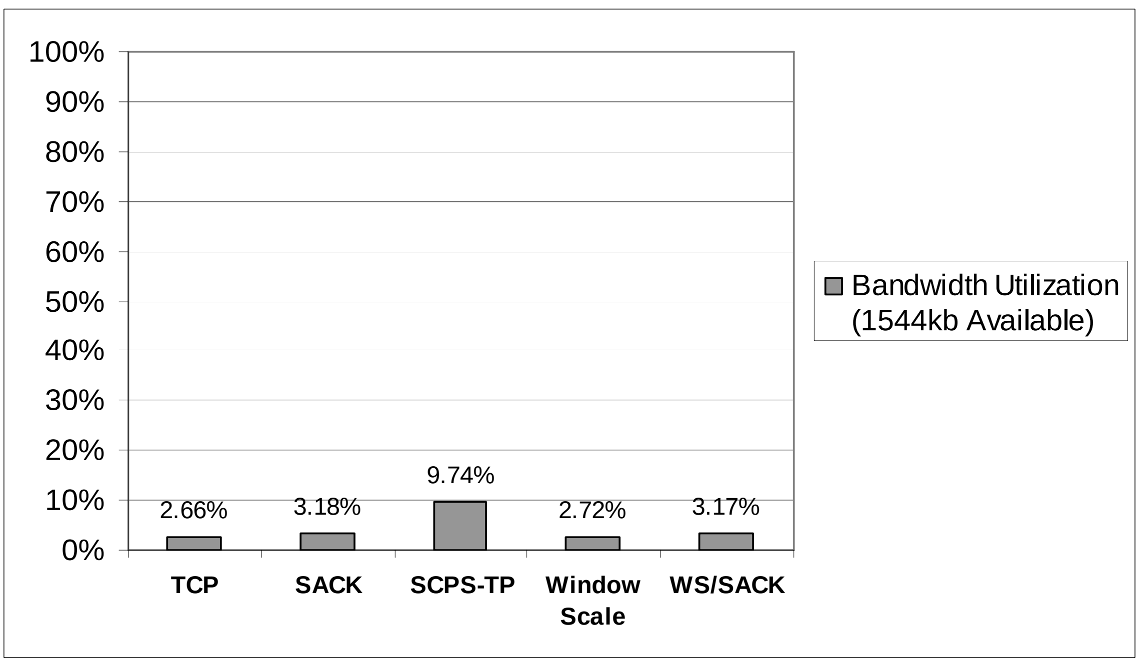

Figure 10. Bandwidth Utilization for 10^{-5} BER/512kb Packet Scenario

Scenario 3: 10^{-6} BER and 128kb packets

As expected, the Kruskal-Wallis test (with a ρ-value less than .0001) revealed a statistical difference exists in the probability distributions for each of the enhancement techniques. The results of this test can be seen in Table 15.

Table 15. Kruskal-Wallis Test Results for 10^{-6} BER/128kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	1722.5	57.42
SCPS-TP	30	3980	132.67
TCP (No Enhancement)	30	1541	51.37
Window Scale	30	2346	78.2
Window Scale/SACK	30	1735.5	57.85

Knowing that a statistical difference exists between at least one pair of the enhancement methods, we can now rely upon descriptive statistics to determine the differences. A 95% confidence interval was calculated for each of enhancement methods with respect to throughput. The results of these calculations can be seen in Table 16.

Table 16. Descriptive Statistics for 10^{-6} BER/128kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	180.6	10.8	116.59	180.6 ± 4.03
SCPS-TP	30	401.63	70.63	4988.17	401.63 ± 26.37
TCP (Unmodified)	30	177.9	12.17	148.16	177.9 ± 4.55
Window Scale	30	188.9	14.11	199.13	188.9 ± 5.27
Window Scale/SACK	30	181.03	14.58	212.52	181.03 ± 5.44

From Table 16, it can be observed that the confidence intervals somewhat reveal the groupings of the various enhancement techniques. Again, SCPS-TP can be placed in its own group, as no other method comes close to its performance in terms of throughput. The window scale strategy does have some overlap with respect to the remaining techniques, but could marginally be considered in its own group. The remaining modifications (SACK, window scale/SACK, TCP) comprise the final grouping. The percentage of improvement to throughput, based on the TCP baseline, can be seen in Figure 11, as well as the bandwidth utilization in Figure 12.

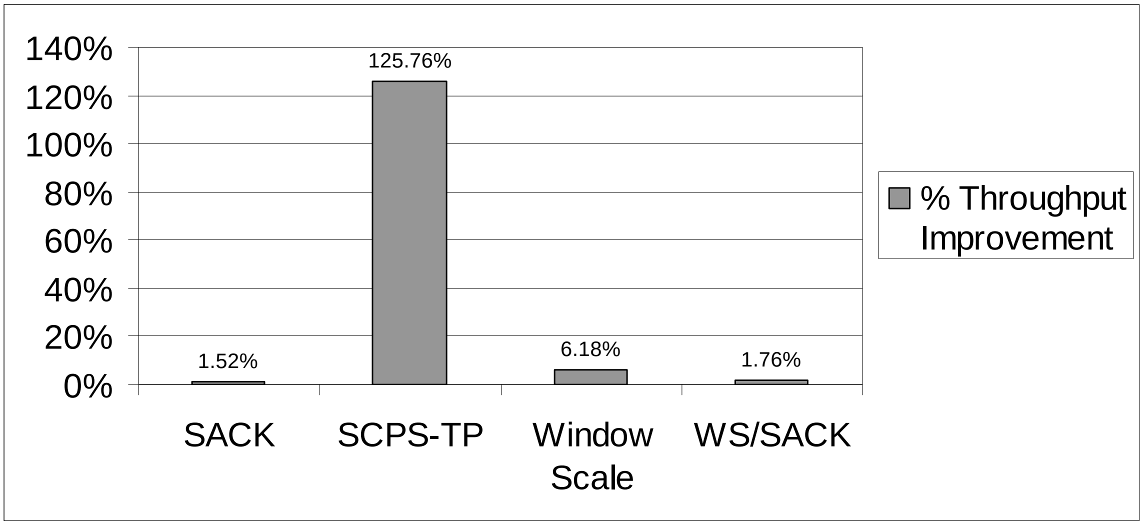

Figure 11. Throughput Improvement Percentages for 10-6 BER/128kb Packet Scenario

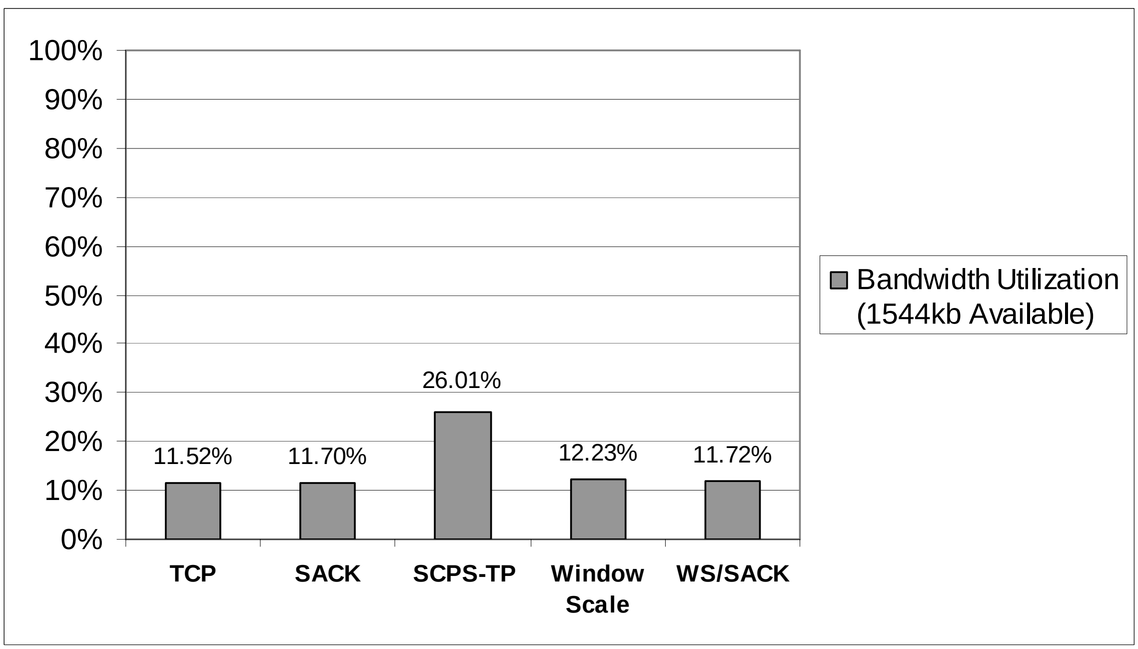

Figure 12. Bandwidth utilization for 10^{-6} BER/128kb Packet Scenario

Scenario 4: 10^{-6} BER and 512kb packets

The execution of the Kruskal-Wallis test highlighted (with a ρ-value less than .0001) the fact that a statistical difference exists between the probability distributions for each of the enhancement techniques. The results of this test can be seen in Table 17.

Table 17. Kruskal-Wallis Test Results for 10^{-6} BER/512kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	1933.5	64.45
SCPS-TP	30	3945	131.5
TCP (No Enhancement)	30	1268.5	42.28
Window Scale	30	1792	59.73
Window Scale/SACK	30	2386	79.53

Now that it is proven that a statistical difference exists between at least one of the enhancement methods, we can now rely upon descriptive statistics to determine the differences, mainly through use of confidence intervals (α = .95). These confidence intervals can be seen in Table 18.

Table 18. Descriptive Statistics for 10^{-6} BER/512kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	182.6	10.45	109.15	182.6 ± 3.9
SCPS-TP	30	398.43	75.63	5719.7	398.43 ± 28.24
TCP (Unmodified)	30	174.73	9.9	98.06	174.73 ± 3.7
Window Scale	30	181.37	15.87	251.9	181.37 ± 5.93
Window Scale/SACK	30	189.57	14.77	218.19	189.57 ± 5.52

From Table 18, the following groupings seem to be present: SCPS-TP makes up one group; the SACK, window scale, and window scale/SACK strategies make up another; finally, TCP is in its own group. The percentage of improvement to throughput, based on the TCP baseline, can be seen in Figure 13. In Figure 14, the amount of bandwidth utilization for each technique is displayed.

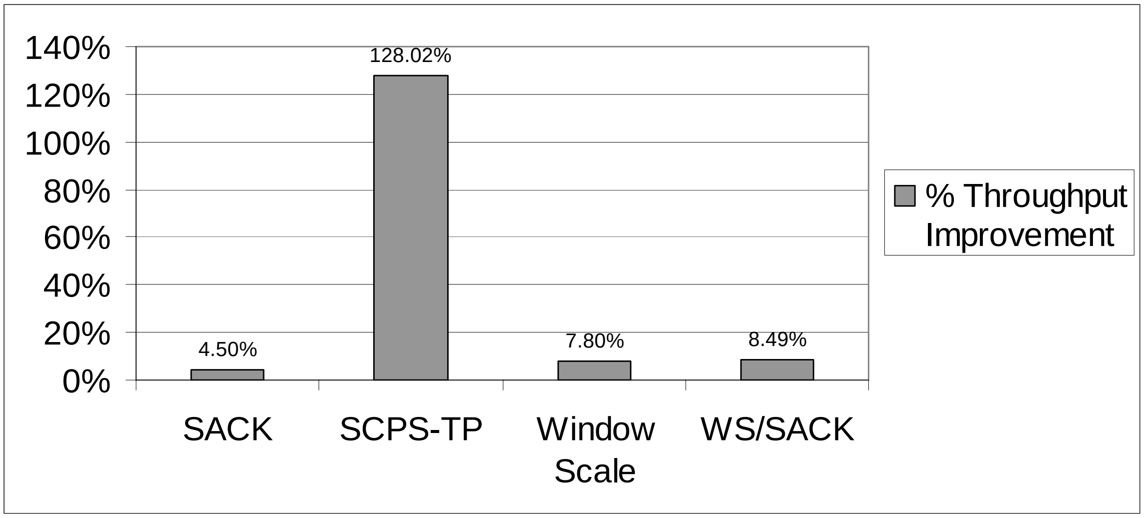

Figure 13. Throughput Improvement Percentages for 10^{-6} BER/512kb Packet Scenario

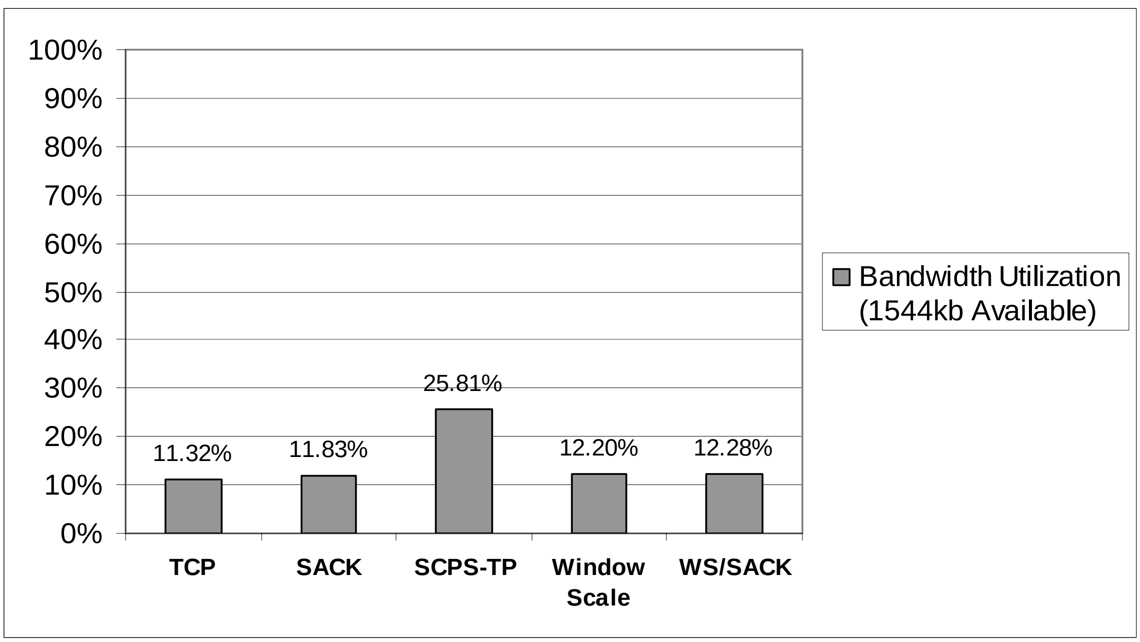

Figure 14. Bandwidth Utilization for 10^{-6} BER/512kb Packet Scenario

Scenario 5: 10^{-7} BER and 128kb packets

The failure of validating the assumptions of AOV again required the use of a non-parametric method. Execution of this method resulted in a ρ-value less than .0001, indicating that a statistical difference exists between the various enhancement techniques. The results of this test can be seen in Table 19.

Table 19. Kruskal-Wallis Test Results for 10^{-7} BER/128kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	1273	42.43
SCPS-TP	30	3945	131.5
TCP (No Enhancement)	30	617	20.57
Window Scale	30	2390	79.67
Window Scale/SACK	30	3100	103.33

Given that a statistical difference exists between at least two of the enhancement methods, confidence intervals ($\alpha = .95$) can now be generated to further explain the respective differences in performance. These confidence intervals are tabulated in Table 20 below.

Table 20. Descriptive Statistics for 10^{-7} BER/128kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	382.47	9.66	93.36	382.47 ± 3.61
SCPS-TP	30	1157.87	174.16	30330.74	1157.87 ± 65.03
TCP (Unmodified)	30	359.03	17.06	290.99	359.03 ± 6.37
Window Scale	30	538.97	55.14	3040.38	538.97 ± 20.59
Window Scale/SACK	30	621.03	35.98	1294.24	621.03 ± 13.43

Table 20 seems to indicate, based on the calculated confidence intervals, that each of the techniques should be grouped individually. The percentage of improvement to throughput, with respect to the unmodified TCP method, can be seen in Figure 15, with bandwidth utilization shown in Figure 16.

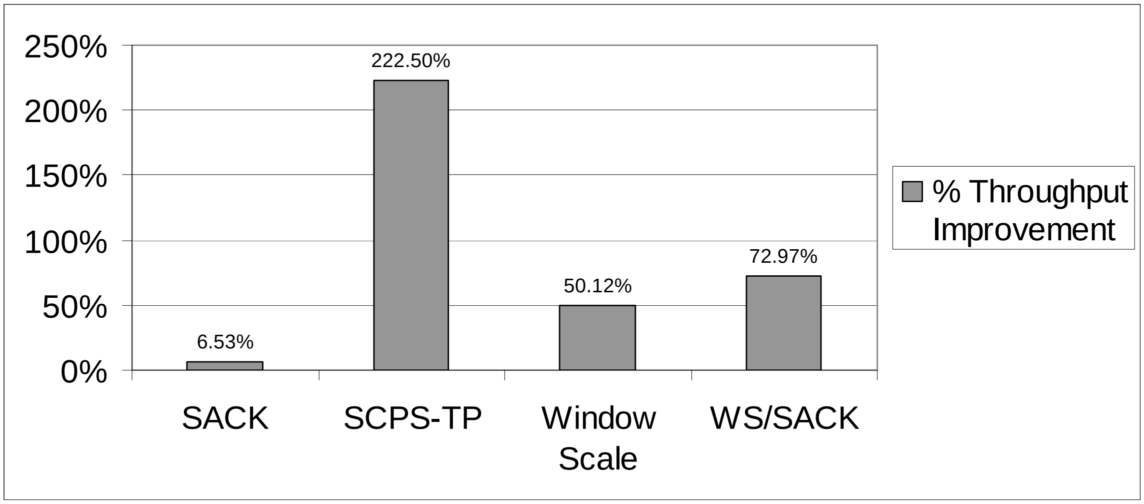

Figure 15. Throughput Improvement Percentages for 10^{-7} BER/128kb Packet Scenario

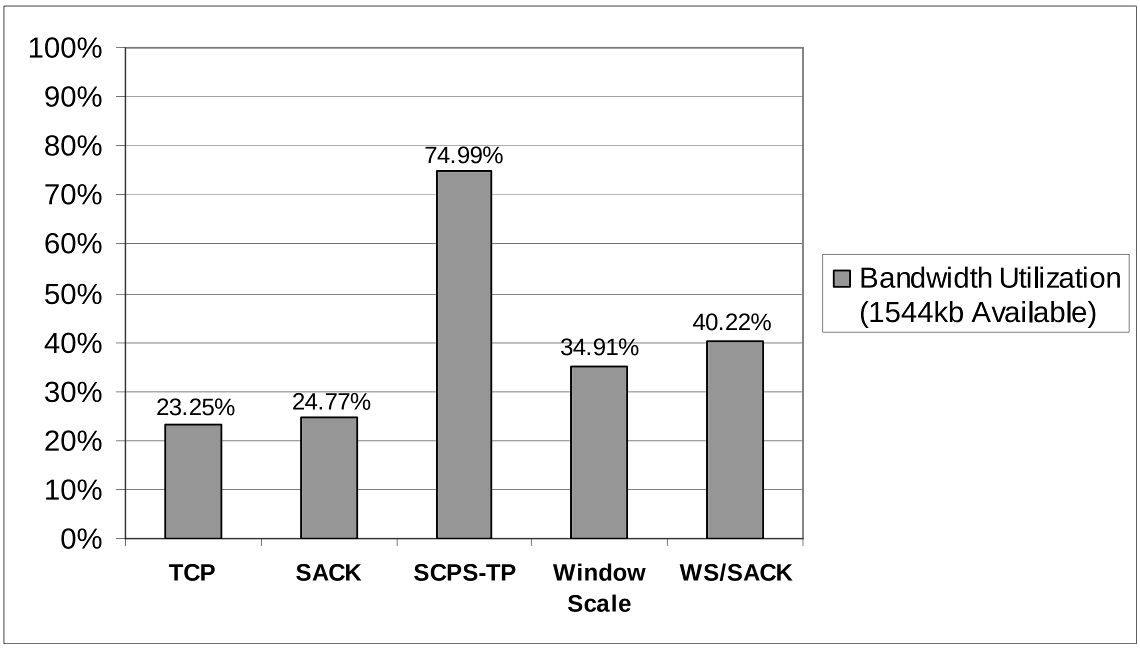

Figure 16. Bandwidth Utilization for 10^{-7} BER/128kb Packet Scenario

Scenario 6: 10^{-7} BER and 512kb packets

The results of the Kruskal-Wallis test indicated that a statistical difference exists between at least two of the enhancement techniques. The results of this test can be seen in the table below.

Table 21. Kruskal-Wallis Test Results for 10^{-7} BER/512kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	1273	42.43
SCPS-TP	30	3945	131.5
TCP (No Enhancement)	30	617	20.57
Window Scale	30	2418.5	80.62
Window Scale/SACK	30	3071.5	102.38

From the score means, it can be seen that a difference exists between the different methods of TCP improvement. Knowing this, confidence intervals ($\alpha = .95$) can be determined to ascertain what these differences are with respect to throughput performance (see Table 22).

Table 22. Descriptive Statistics for 10^{-7} BER/512kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	379.83	7.95	63.25	379.83 ± 2.97
SCPS-TP	30	1153.1	167.43	28033.4	1153.1 ± 62.52
TCP (Unmodified)	30	357.5	18.03	324.95	357.5 ± 6.73
Window Scale	30	536.93	51.7	2672.69	536.93 ± 19.3
Window Scale/SACK	30	604.7	29.08	845.46	604.7 ± 10.86

The above table shows that none of the enhancement techniques overlap with respect to a 95% confidence interval. As such, each of the improvement methods should not be grouped with another in terms of throughput performance; they do not perform in a similar enough fashion to warrant any type of grouping. Throughput performance improvements and bandwidth utilization for each of the methods in consideration are shown in Figures 17 and 18.

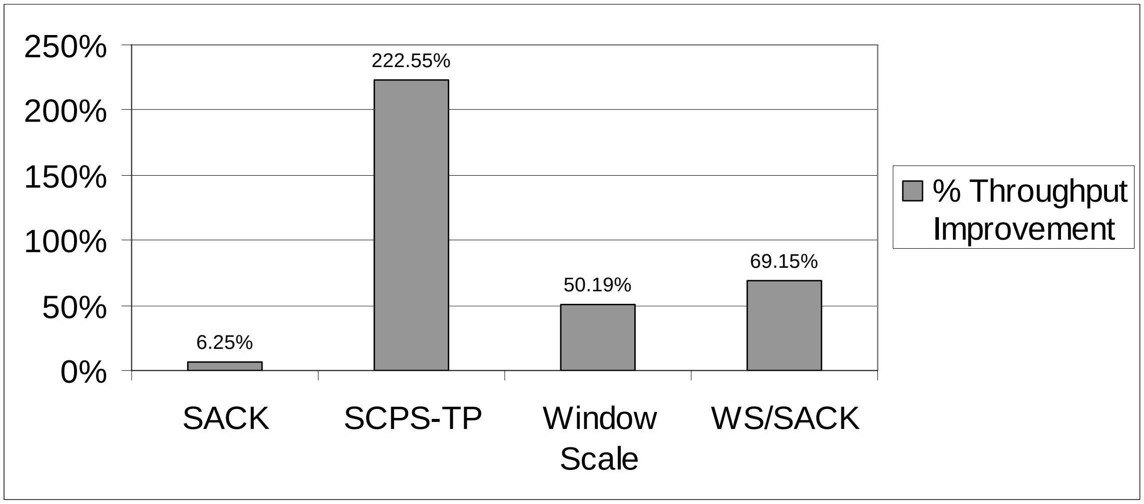

Figure 17. Throughput Improvement Percentages for 10^{-7} BER/512kb Packet Scenario

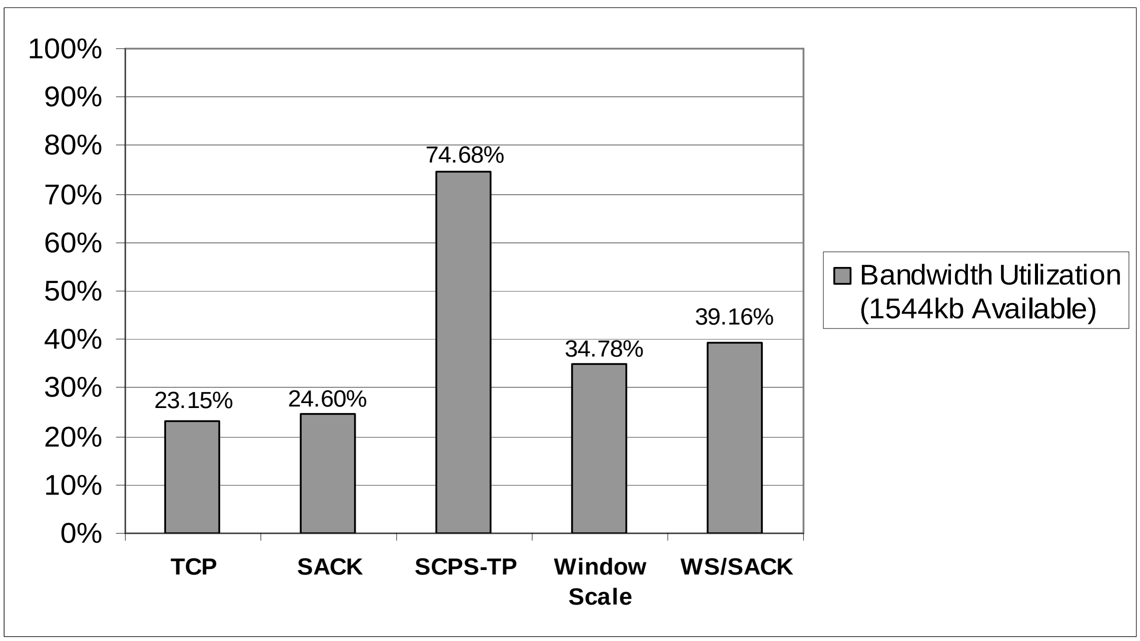

Figure 18. Bandwidth Utilization for 10^{-7} BER/512kb Packet Scenario

Scenario 7: 10^{-8} BER and 128kb packets

The results of the Kruskal-Wallis test also indicated that a statistical difference (with a (ρ-value less than .0001) exists between at least two of the enhancement techniques in this scenario. The results of this test can be seen in the Table 23.

Table 23. Kruskal-Wallis Test Results for 10^{-8} BER/128kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	769	25.63
SCPS-TP	30	4065	135.5
TCP (No Enhancement)	30	1061	35.37
Window Scale	30	3165	105.5
Window Scale/SACK	30	2265	75.5

From the scores in Table 23, it can be observed that a difference exists between the differing methods of TCP enhancement. With this knowledge, confidence intervals can be calculated to further determine what these differences are in terms of throughput performance (see Table 24).

Table 24. Descriptive Statistics for 10^{-8} BER/128kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	421.57	3.27	10.67	421.57 ± 1.22
SCPS-TP	30	1426.1	8.72	76.02	1426.1 ± 3.26
TCP (Unmodified)	30	424.03	5.2	26.99	424.03 ± 1.94
Window Scale	30	901.07	0.25	0.06	901.07 ± 0.09
Window Scale/SACK	30	830.07	18.5	342.34	830.07 ± 6.91

The above table shows that only the SACK and generic TCP confidence intervals overlap; all other methodologies experience no type of convergence. Therefore, each of the improvement methods, except for TCP and SACK, should not be grouped with another method in terms of throughput performance. TCP and SACK perform in a statistically equivalent fashion in this scenario and there should be grouped together. Throughput performance, in terms of improvement with respect to standard TCP, can be seen in Figure 19; bandwidth utilization is captured in Figure 20.

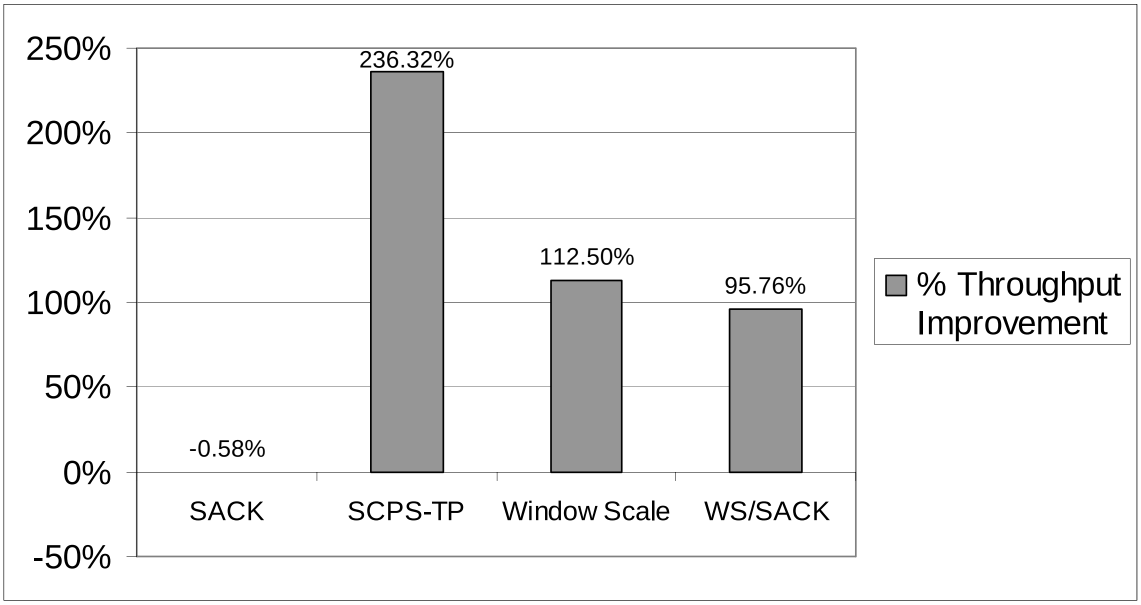

Figure 19. Throughput Improvement Percentages for 10^{-8} BER/128kb Packet Scenario

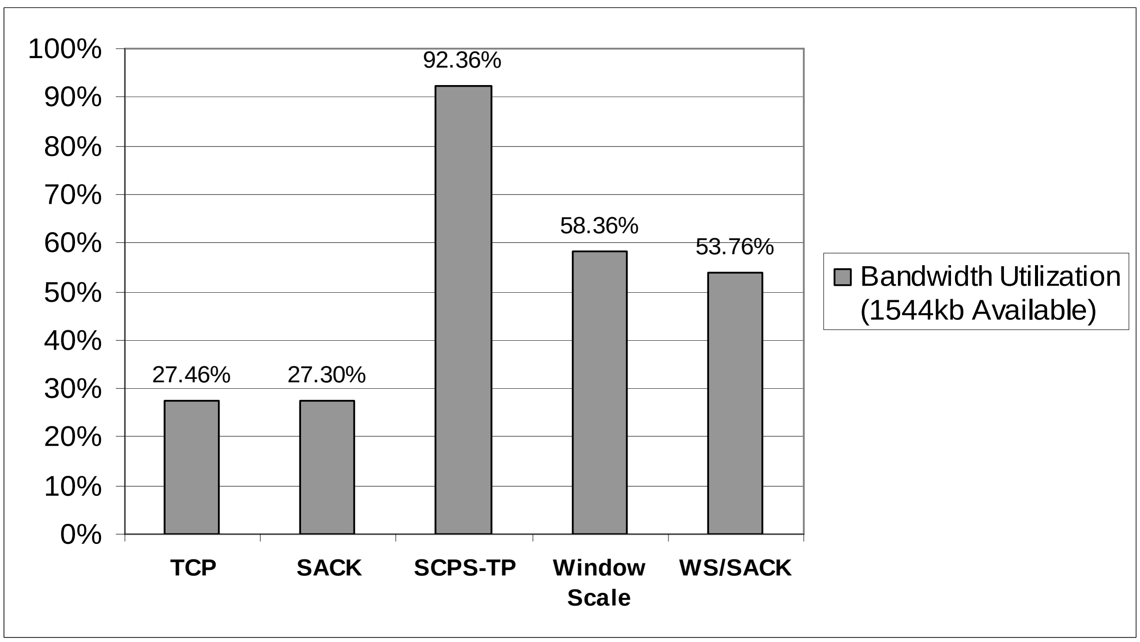

Figure 20. Bandwidth Utilization for 10^{-8} BER/128kb Packet Scenario

Scenario 8: 10^{-8} BER and 512kb packets

The results of the non-parametric test, Kruskal-Wallis, confirmed that a statistical difference, at a ρ-value less than .0001, exists between at least two of the enhancement techniques based on the rank sum score means. A summary of the Kruskal-Wallis test can be seen below.

Table 25. Kruskal-Wallis Test Results for 10^{-8} BER/512kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	709.5	23.65
SCPS-TP	30	4065	135.5
TCP (No Enhancement)	30	1120.5	37.35
Window Scale	30	3165	105.5
Window Scale/SACK	30	2265	75.5

Based on the results in Table 25, it will be necessary to determine confidence intervals for the various enhancement methodologies in order to ascertain the differences between the different approaches. These calculations can be seen in the table below.

Table 26. Descriptive Statistics for 10^{-8} BER/512kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	422.57	3.59	12.87	422.57 ± 1.34
SCPS-TP	30	1425.478	9.22	85.02	1425.47 ± 3.44
TCP (Unmodified)	30	425.73	4.03	16.2	425.73 ± 1.5
Window Scale	30	903.97	0.19	0.03	903.97 ± 0.03
Window Scale/SACK	30	835.84	20.89	434.47	835.84 ± 7.67

The above table shows that none of confidence intervals overlap, however, the SACK and generic TCP techniques are extremely close in performance concerning throughput, making it reasonable to assume that their throughput performance results are equivalent. The remaining methodologies provide significant gains when compared to TCP; of the three, SCPS-TP performed at the highest level, with window scale second, and the window scale/SACK solution placing third. The percentage of throughput improvement can be seen in Figure 21, with bandwidth utilization in Figure 22.

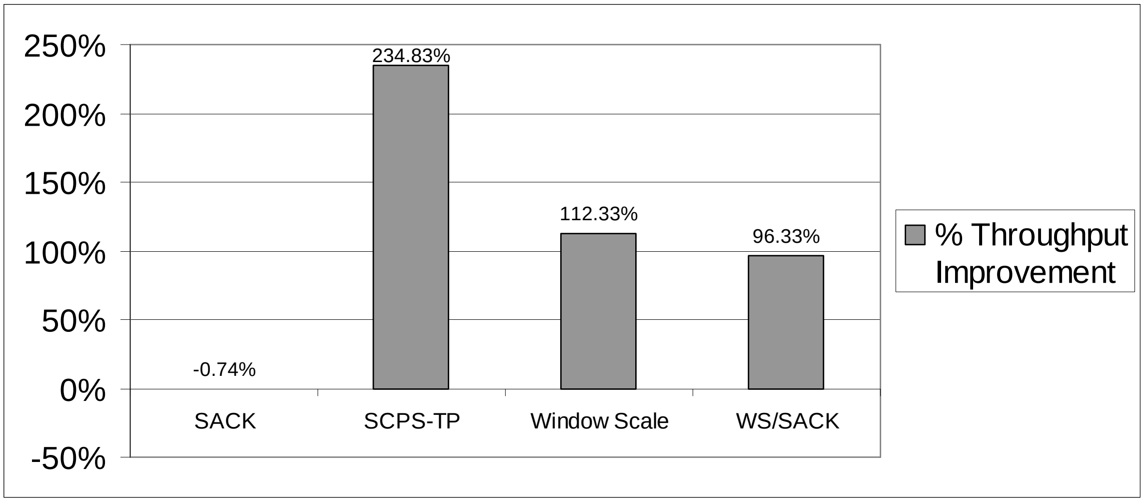

Figure 21. Throughput Improvement Percentages for 10^{-8} BER/512kb Packet Scenario

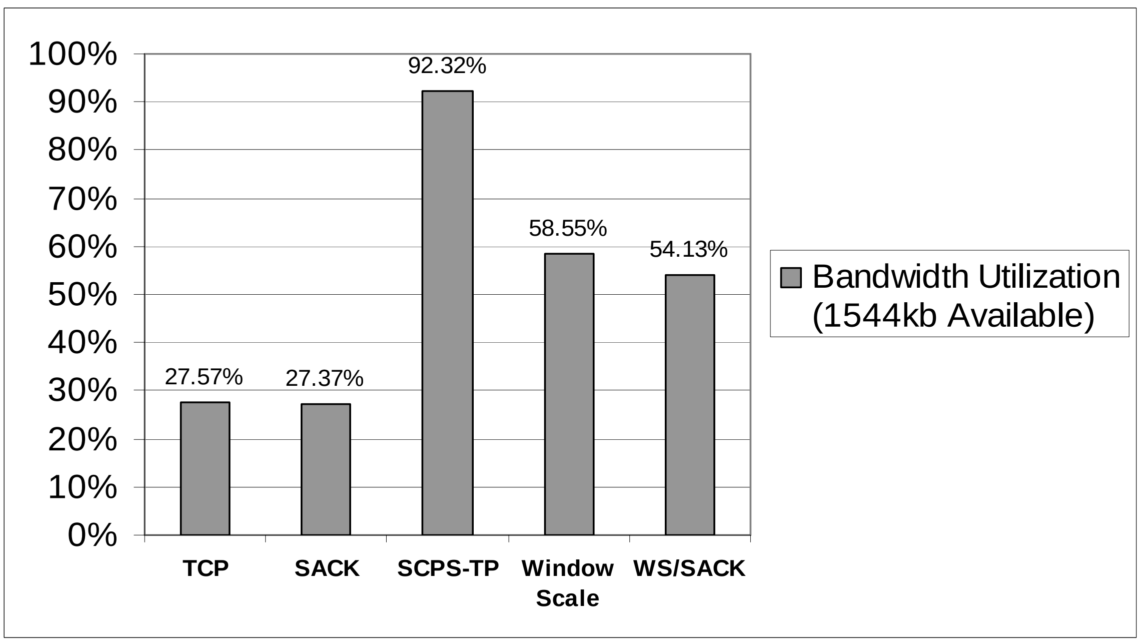

Figure 22. Bandwidth Utilization for 10^{-8} BER/512kb Packet Scenario

Scenario 9: 10^{-9} BER and 128kb packets

Failures to confirm the AOV model assumptions indicated the need for a non-parametric method. Execution of the Kruskal-Wallis test confirmed that a disparity exists between at least two of the enhancement methodologies (at a ρ-value less than .0001). A summary of the Kruskal-Wallis test can be seen in Table 27.

Table 27. Kruskal-Wallis Test Results for 10^{-9} BER/128kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	780	26.0
SCPS-TP	30	4005	133.5
TCP (No Enhancement)	30	1050	35.0
Window Scale	30	2760	92.0
Window Scale/SACK	30	2730	91.0

Based on these non-parametric results, confidence intervals were generated to determine the differences between the various techniques. The results of these calculations are displayed in the table below.

Table 28. Descriptive Statistics for 10^{-9} BER/128kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	430.8	1.45	2.1	430.8 ± 0.54
SCPS-TP	30	1414.07	164.14	26943.37	1414.07 ± 61.29
TCP (Unmodified)	30	431.7	1.51	2.29	431.7 ± 0.56
Window Scale	30	901.1	0.31	0.09	901.1 ± .11
Window Scale/SACK	30	901.07	0.25	0.06	901.07 ± 0.09

The above table shows a significant amount of overlap between two pairs of techniques. First, the window scale and window scale/SACK strategies appear to provide an equivalent level of performance benefit. Second, the TCP and SACK methods also seem to provide an equivalent amount of throughput performance. The only enhancement not equivalent to any of the others is SCPS-TP, as it performs well beyond any of the other methods. The percentage of throughput improvement afforded by all of these techniques, with respect to standard TCP, can be seen in Figure 23. Bandwidth utilization is shown in Figure 24.

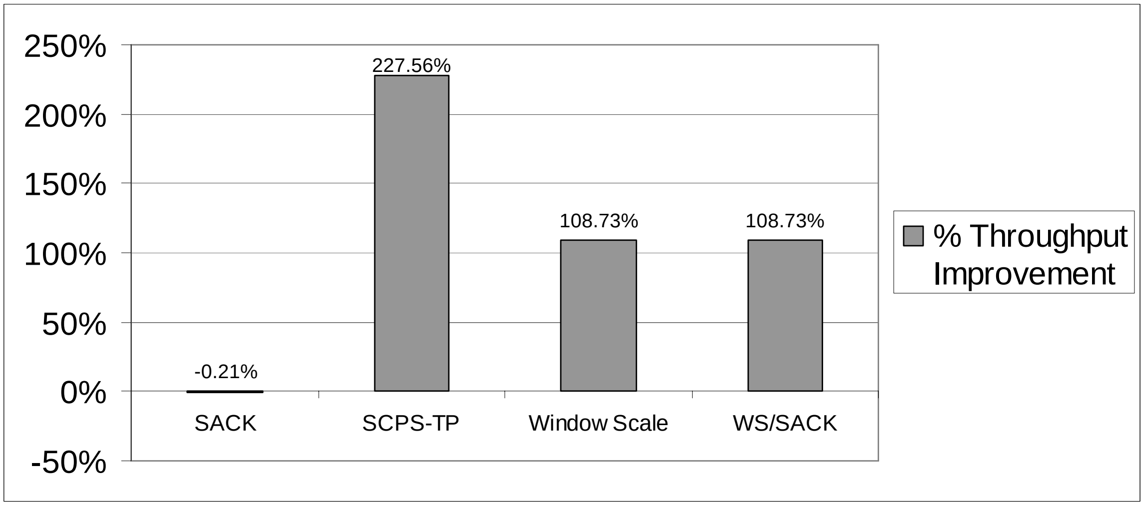

Figure 23. Throughput Improvement Percentages for 10^{-9} BER/128kb Packet Scenario

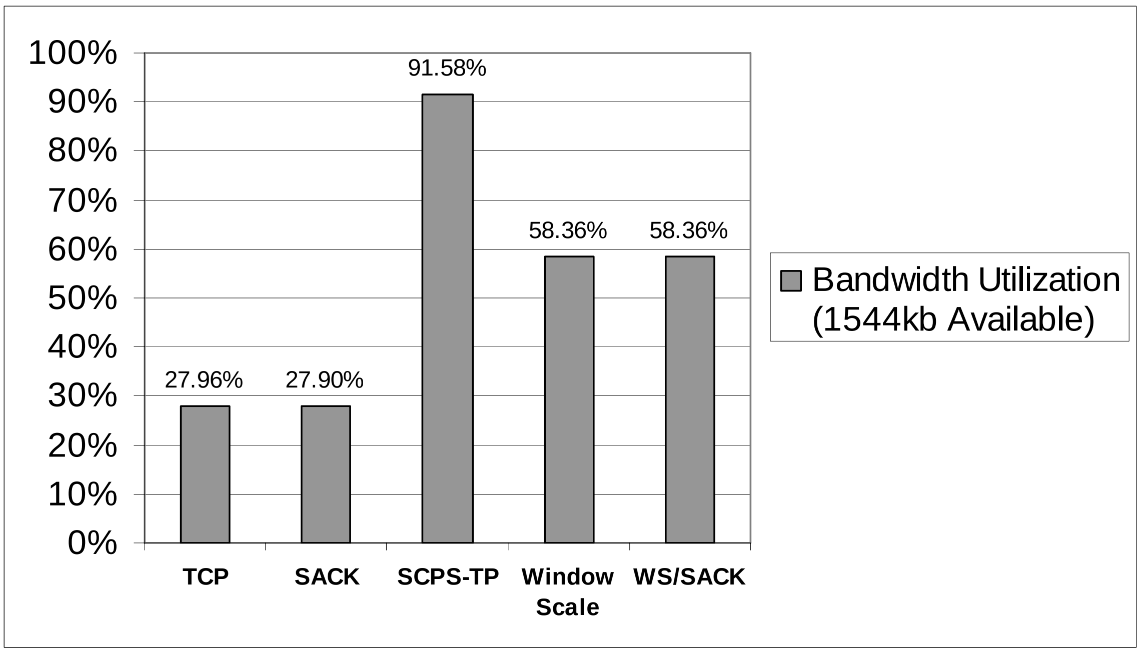

Figure 24. Bandwidth Utilization for 10-9 BER/128kb Packet Scenario

Scenario 10: 10^{-9} BER and 512kb packets

The Kruskal-Wallis method was also used for this scenario, with results (ρ-value less than .0001) indicating a difference exists between at least two of the throughput improvement techniques. A summary of the Kruskal-Wallis test can be seen in Table 29.

Table 29. Kruskal-Wallis Test Results for 10^{-9} BER/512kb Packet Scenario

Enhancement Technique	Count	Score Sum	Score Mean
SACK	30	970	32.33
SCPS-TP	30	4065	135.5
TCP (No Enhancement)	30	860	28.67
Window Scale	30	2715	90.5
Window Scale/SACK	30	2715	90.5

As previously explain, Kruskal-Wallis indicates differences exist, but not was the differences between samples are in terms of a mean or other descriptive method. As such, confidence intervals were generated to determine and quantify these. The results can be seen in Table 30.

Table 30. Descriptive Statistics for 10^{-9} BER/512kb Packet Scenario

Enhancement	Count	Mean	Standard Deviation	Variance	Confidence Interval
SACK	30	432.1	2.06	4.23	432.1 ± 0.77
SCPS-TP	30	1442.03	12.58	158.31	1442.03 ± 4.7
TCP (Unmodified)	30	431.73	2.21	4.89	431.73 ± 0.83
Window Scale	30	904	0.0	0.0	904 ± 0.0
Window Scale/SACK	30	904	0.0	0.0	904 ± 0.0

From the above table, it can be seen that overlap occurs between two pairs of techniques; the SACK and TCP methods overlap, as do the window scale and window scale/SACK methods. Again, as in all previous scenarios, the SCPS-TP method differs greatly from all other techniques in terms of transmission throughput. Transmission performance improvements, as compared to TCP, can be seen in Figure 25. Bandwidth utilization results are displayed graphically in Figure 26.

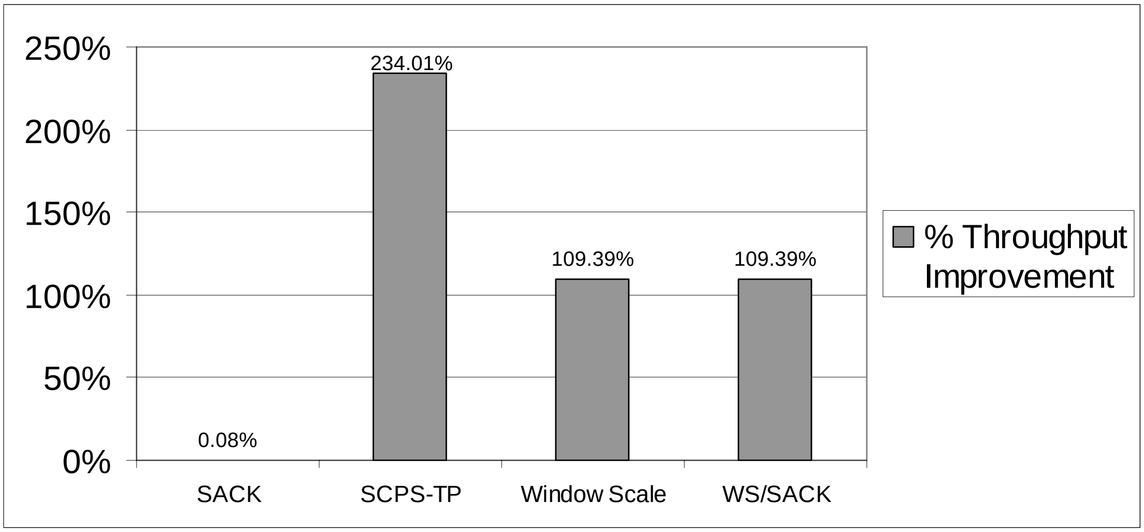

Figure 25. Throughput Improvement Percentages for 10^{-9} BER/512kb Packet Scenario

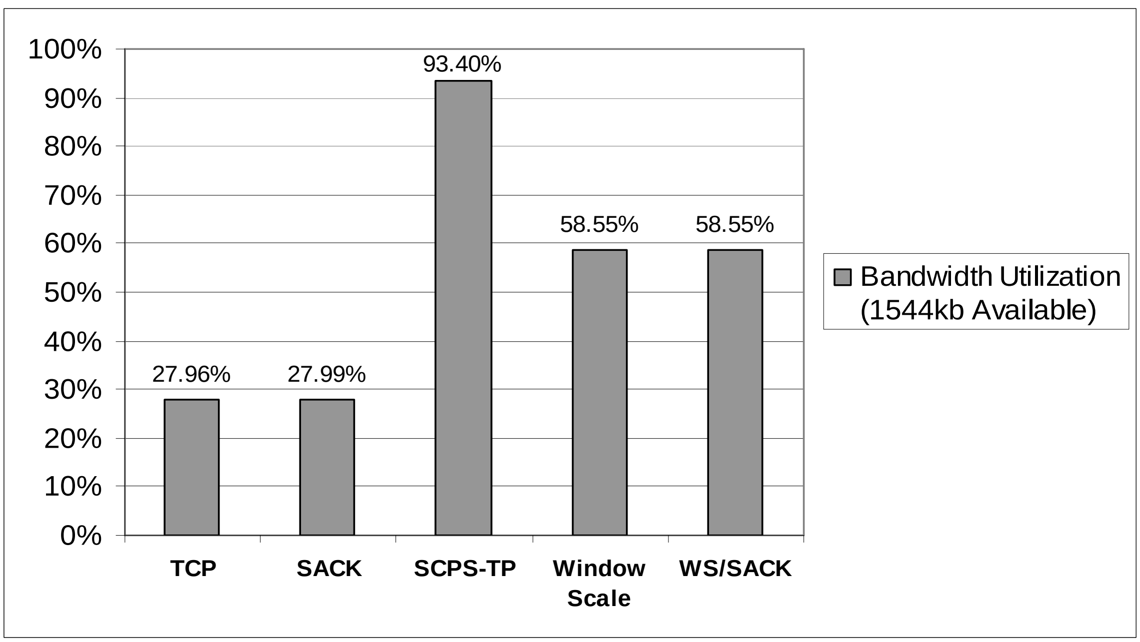

Figure 26. Bandwidth Utilization for 10^{-9} BER/512kb Packet Scenario

Summary of Statistical Findings

It is obvious from the preceding sections that amount of throughput or bandwidth utilization experienced varied depending on the type of TCP technique utilized. In consideration of overall performance, SCPS-TP presented the greatest amount of gain, in terms of both throughput and bandwidth utilization, regardless of the transmission error rate or packet size utilized. The other methods did begin to differentiate themselves from one another until testing began at an error rate of 10^{-7}. At this error rate, the window scale and window scale/SACK techniques began giving significantly greater gains for throughput and bandwidth utilization than standard TCP. Finally, the SACK implementation either performed at a level that was consistent with performance numbers found with a standard TCP implementation or, at times, performed at level lower than generic TCP. These findings are summarized below in Figures 27, 28, 29, and 30.

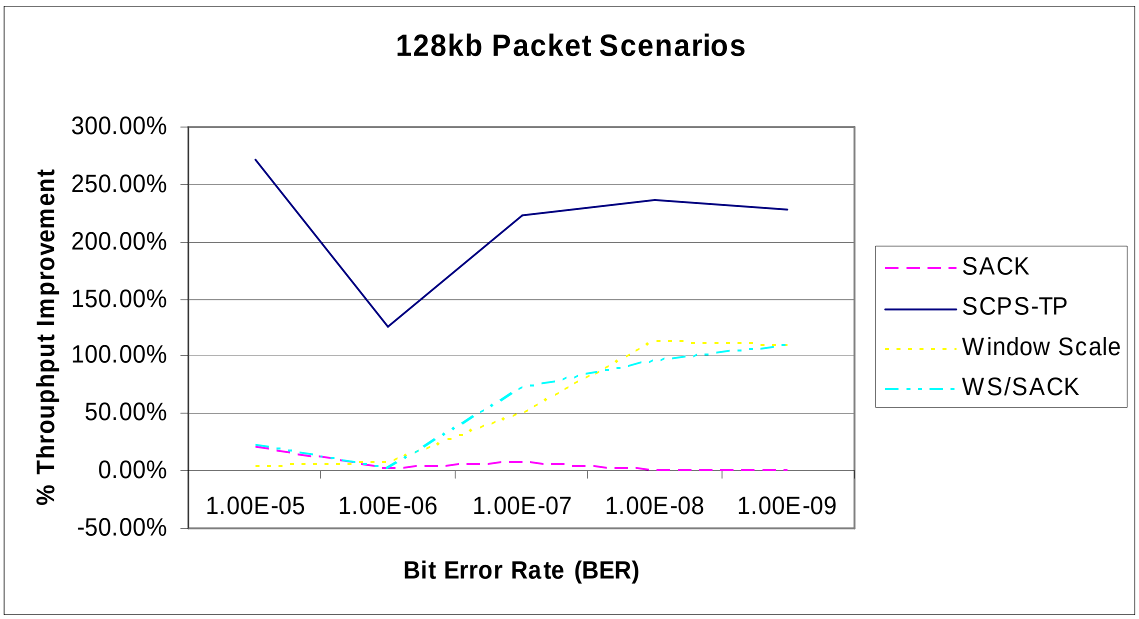

Figure 27. Throughput Improvement Percentages for 128kb Packet Scenarios

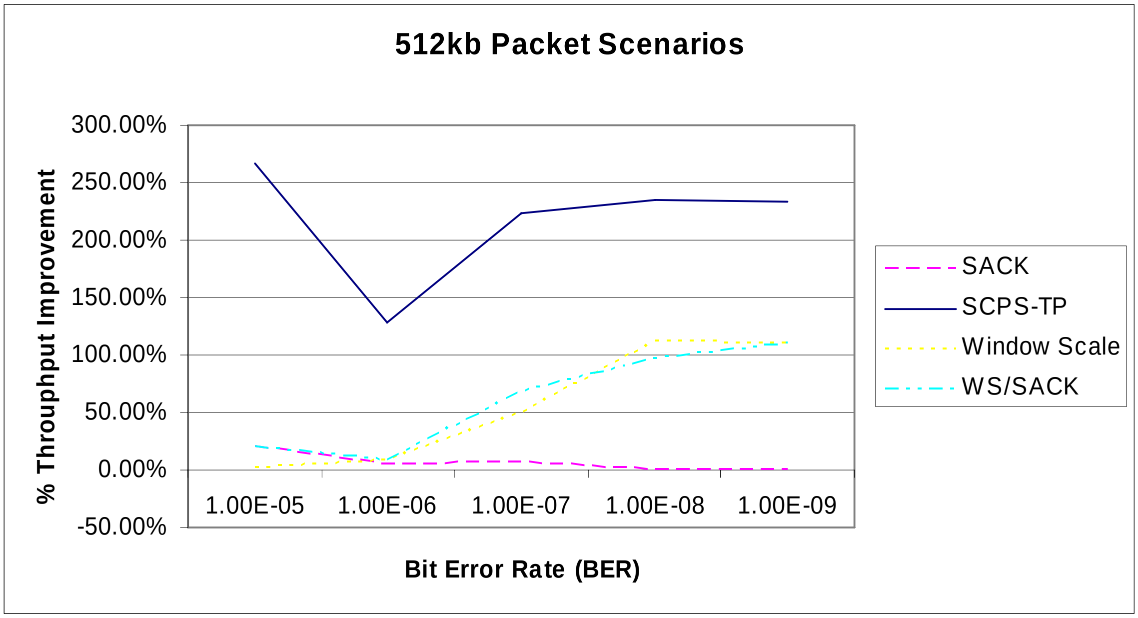

Figure 28. Throughput Improvement Percentages for 512kb Packet Scenarios

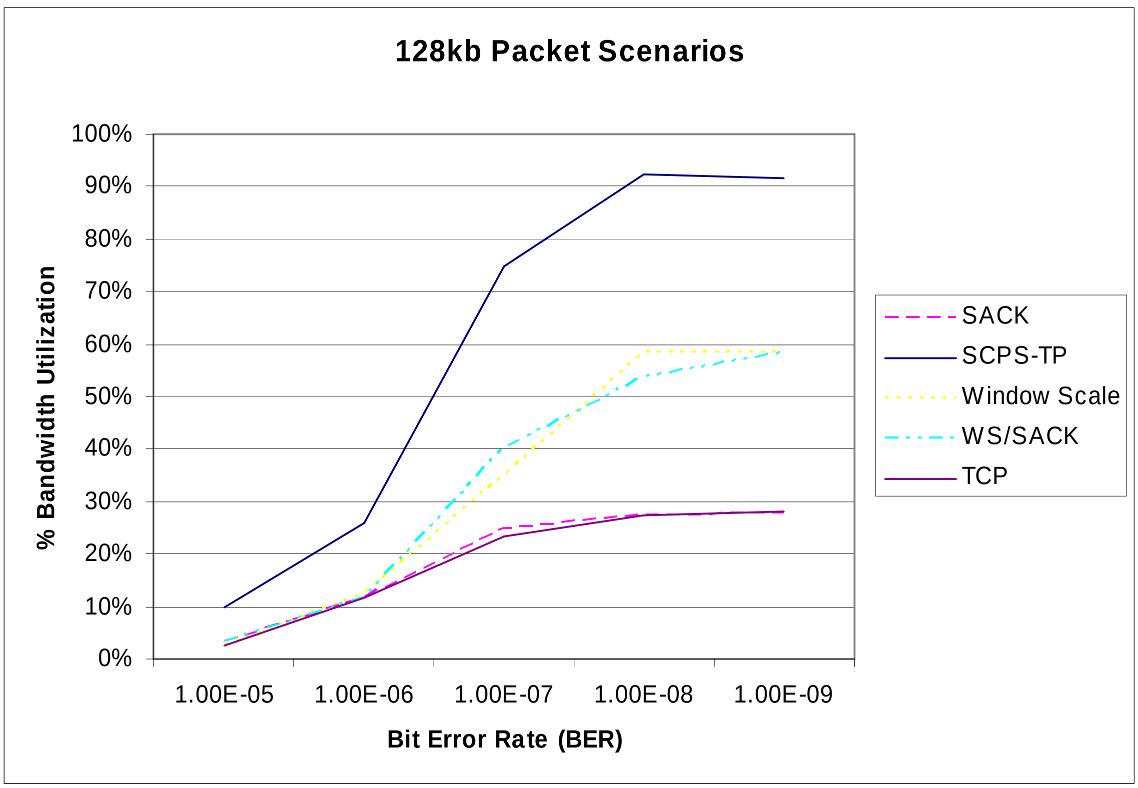

Figure 29. Bandwidth Utilization for 128kb Packet Scenarios

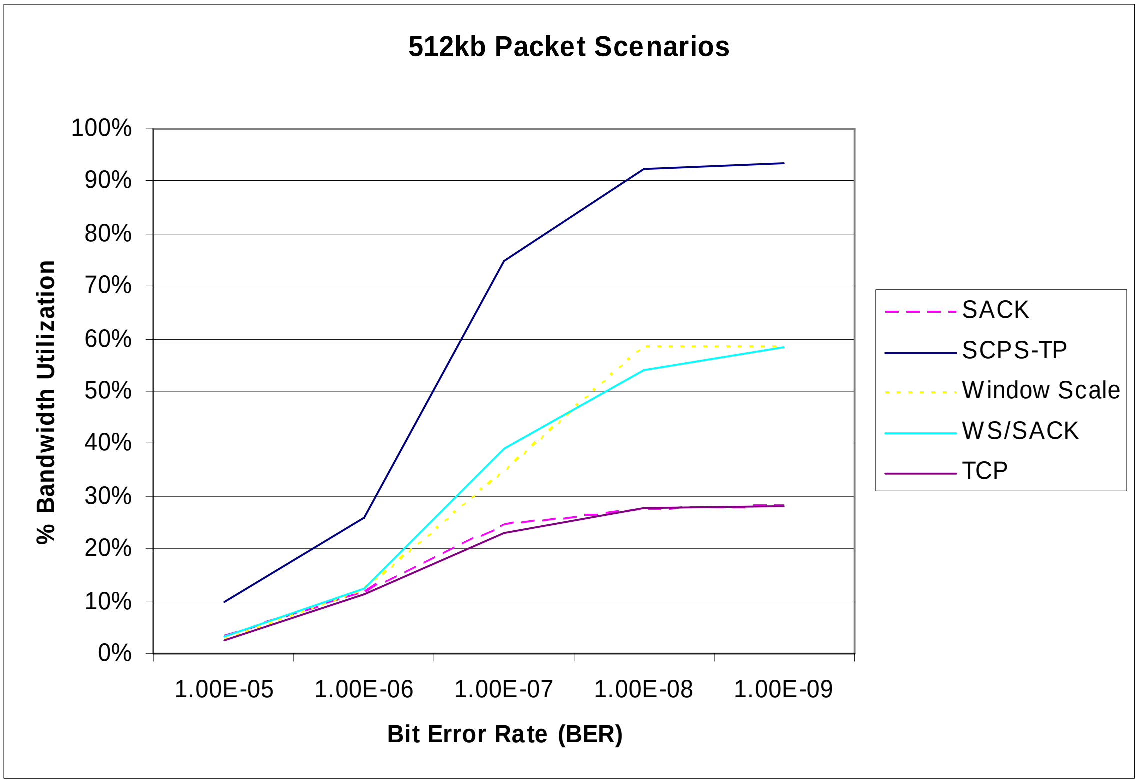

Figure 30. Bandwidth Utilization for 512kb Packet Scenarios

Chapter Summary

This chapter presented an overview of the statistical analyses for this study. Items presented in this chapter included the individual scenario results for TCP enhancement strategies and an overview of the overall statistical findings of this work. The next chapter will present a summation of these findings and will use them to answer the research questions posed in Chapter 1.

V. Conclusions and Recommendations

Chapter Overview

This chapter will present the answers to the research questions posited in Chapter 1, as well as provide the research conclusions, in terms of business case justifications for the various enhancement methodologies, for this study. In addition future research recommendations will also be presented.

Answers to Research Questions

***Research Question 1: What is the performance benefit (in terms of throughput) of implementing a SACK enhancement to TCP when TCP is utilized over a satellite transmission system*?**

From above, it can be seen that RQ1 mainly dealt with the determination of the performance benefits of the various TCP enhancement mechanisms when utilized over a satellite transmission system. In particular, RQ1 sought for the performance benefits of the SACK technique in a satellite communication system. From the analysis in the previous chapter, it was apparent that the SACK schema provided maximum benefit (in terms of its performance over the various scenarios) when used during the highest level of BER (10^{-5}); depending the packet size transmitted, the SACK technique provided up to a 20.64% (128kb packets, with a 19.4% improvement experienced with 512kb packets) improvement as compared to a standard TCP implementation. This improvement rate, however, was not experienced at the other treatment levels of BER, as the SACK solution performed at levels comparable to standard TCP. The spanner program utilized in the experimentation makes use of a Bernoulli distribution for bit error occurrence; the likelihood of experiencing a bit error is merely a function of the assigned BER and the time period of testing. It is likely that this property limited the effectiveness of the SACK technique, as compared to generic TCP, since not enough errors occurred at the higher BERs with respect to the test timeline of 300 seconds, which mitigated the need for a mechanism to prevent the execution of the congestion control algorithms. In addition, real world satellite communication systems typically experience bit errors in bursts, whereas the spanner program, with use of a Bernoulli distribution, will insert errors in a somewhat systematic pattern. It is likely that the SACK solution would have provided greater efficiency, in terms of throughput and as compared to TCP, if bursts of errors were experienced, as this technique could act in a single RTT and belie the need for execution of the TCP congestion control algorithms.

Research Question 2: What is the performance benefit of implementing a window scale modification to TCP when TCP is utilized over a satellite transmission system?

The second research question, RQ2, dealt with the determination of the performance benefits of the window scale solution when used in a satellite transmission system. At the highest BER levels (10^{-5} and 10^{-6}), the window scale technique provided throughput performance levels that were comparable to a standard TCP solution. This result is somewhat intuitive, as probability of a bit error occurring (at the specified test duration) is likely and would result in significantly slower growth of the TCP window, with no advantage for the fast start or larger window properties of the window scale solution. At the lower BER levels, the window scale method provided significant throughput performance levels, when compared to TCP, with minimum increase of 50.12% and a high of 112.5%. Again, the likelihood of error occurrence is significantly decreased, allowing uninhibited growth of the TCP window; the fast start and larger maximum window properties provide significant advantages over standard TCP mechanisms.

Research Question 3: What is the performance benefit of implementing both a SACK enhancement and a window scale modification to TCP when TCP is utilized over a satellite transmission system?

The next research question, RQ3, asked for the performance benefit of using both the SACK and window scale modifications in a satellite transmission system. From the results of the statistical analyses in Chapter 4, it was obvious that the combinational use of SACKs and window scale provided superior performance to TCP in almost every scenario (up to 109.30%). The 10^{-6} BER scenarios, however, were one area in which the use of both SACKs and window scale did not provided a significant advantage (improvements of 1.76% and 8.49% for both test scenarios) versus use of standard TCP. This may represent a point of diminishing returns for the combinational TCP enhancement of SACKs and window scale, as bit errors in this experimentation were common enough to inhibit window growth, but evenly spread (due to the spanner program's error distribution) to somewhat negate the advantage of the SACK technique's single RTT identification/resolution of lost segments.

Research Question 4: What is the performance benefit of utilizing PEPs to improve TCP sessions that occur over a satellite transmission system?

The fourth research question, RQ4, indicated the need for quantifying the benefits of a PEP system to the throughput performance of a satellite transmission system. The results of the experimentation showed that a PEP-based enhancement, through use of the SCPS-TP standard, to TCP sessions over a high latency/error transmission medium provided a tremendous level of throughput improvement regardless of the treatment levels of independent variables (such as BER or packet size). The PEP solution provided a minimum improvement to throughput, as compared to TCP, of 125.76%; the upper bound of improvement experienced during the experimentation was 270.72%. In comparison to the test results of the other enhancement techniques, the PEP-based solution provided, at a minimum, twice the performance benefit when compared to standard TCP.

Answers to Research Question 5 Are there statistical differences between the respective TCP enhancements? If so, how can costs (in terms of acquisition and support requirements) be expressed as a decision point for execution of a given TCP enhancement?

The final research question, RQ5, looks for the statistical differences between the TCP enhancement methods and for a cost/benefit analysis. In all of the scenarios, the non-parametric tools used for analysis proved that a statistical difference existed between at least two of the enhancement methods with respect to throughput. Groupings, at a 95% confidence level, were made via use of descriptive statistics (with respect to throughput) to highlight which techniques differed and which performed at the same level. In all of the scenarios, the PEP-based solution performed a significantly different level than the other methods. As stated previously, the PEP approach also outperformed the other methods, making it the TCP enhancement solution of choice in consideration of throughput performance. The remaining techniques did not vary greatly at the highest levels of BER; though it was reasonable to group the SACK and window scale/SACK techniques in one group and window scale and TCP in another, neither group had a large difference between them in terms of throughput. At the next level of BER (10^{-6}), the groupings varied depending on the packet size utilized; the differences between the groupings, again, were not at a significantly different level. At the remaining levels of BER, however, the techniques began to vary greatly. The window scale and window scale/SACK techniques operated at levels greater than SACK or TCP, but less than the PEP solution. The differences in performance between the window scale and window scale/SACK solutions did not vary greatly, making them second in terms of throughput performance to the SCPS-TP PEP. Finally, at the remaining levels of BER, the SACK technique did not perform at levels much different than standard TCP, making its placement as the last alternative when compared to the other enhancement's throughput performance.

In making a business case for any of the methods, it should be obvious that cost will be the overriding factor for any organization in terms of selection for use. In this respect, it will be necessary to understand the lease costs of a satellite channel/transponder. In the author's experience, lease costs not only come from the organization that owns/operates a given space asset, but can also come from entities that govern frequency allocations/rights. Typically termed landing rights, host nations for which a satellite communication link exists will typically require a tariff for use of a piece of the frequency spectrum. In the Middle East and Europe, it was not uncommon for costs for a 1.544Mb/s link to exceed $1M per annum (in extreme cases, per month) when lease costs from the owner's of the spacecraft were combined with the payment due for host nation landing rights. It is therefore recommended that bandwidth utilization should be high in order to maximize return on investment. Obviously, a balancing act must occur with respect to satellite lease costs and costs for implementation of techniques used to maximize bandwidth utilization. This balance could be represented by the inequality:

$$S \times T \times ((BE/100) - (BU/100)) \geq C_{PE} + (C_A \times T) \quad (6)$$

where

S = satellite lease costs, US dollars per unit time

T = unit of time

BU = percentage of bandwidth utilization without enhancement

BE = percentage of potential bandwidth utilization with enhancement

C_{PE} = cost of acquisition of TCP enhancement method

C_A = costs associated with TCP enhancement method (such as training) per unit time

Equation 6 represents a justification for use of a TCP enhancement technique, in terms of dollars per a specified timeframe. The left hand side of the inequality calculates the satellite lease costs per unit on unutilized bandwidth (based on the difference between the potential gains of the selected enhancement technique versus use of standard TCP) during a specified timeframe. The other side of the inequality determines the costs associated with an enhancement's acquisition and recurring costs. Obviously, anytime the satellite lease costs exceed that of a solution set for TCP enhancement, the enhancement should be acquired and utilized in order to maximize the return on investment concerning the satellite lease. Conversely, if the enhancement solution costs exceed lease costs, the enhancement method should not be acquired.

This equation, however, cannot capture other factors, such as infrastructure support in terms of rack space availability or acceptability of size with respect to a deployment loading limitations. As such, a flow chart has been constructed (see Figure 31 to act as a decision aid, in addition to the above equation.

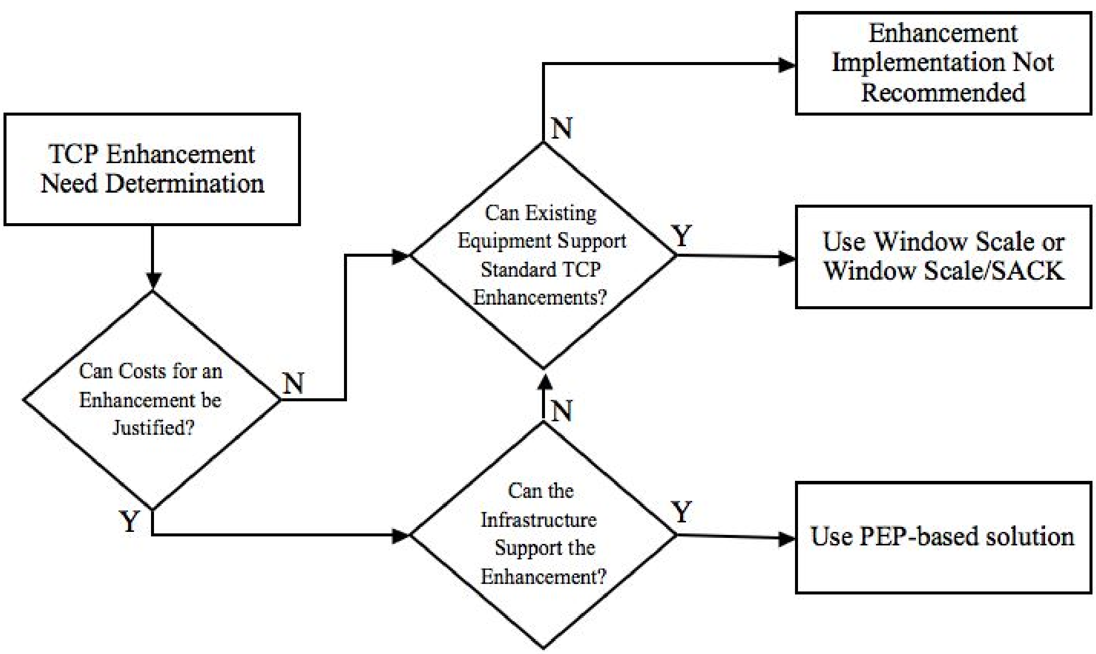

Figure 31. TCP Enhancement Selection Flow Chart

From the above figure, the initial flow chart question is concerned with cost justification. Obviously, if equation 6 is calculated to be favorable and the costs associated with a solution can be covered, the answer for this question will be yes. Following the yes path, the next question asks about infrastructure support. This means that power, rack space, and/or other physical requirements can be met for an enhancement strategy. If yes, it is recommended that a PEP-based solution be employed. If no, the follow up question asks if standard TCP enhancements (SACKs and window scale modifications) can be supported with existing equipment. This question is also posed if the cost question for an enhancement strategy is answered negatively. If the standard mechanisms can be implemented, it is recommended that either a window scale or window scale/SACK solution be implemented. If the answer is no, then no enhancement should be implemented at this time.

Model Validation

The results for the PEP piece of the experimentation were compared to those found in (Durst, Miller, and Travis, 1997) and (Ishac and Allman, 2001); though differing latency levels were used and data transfer mechanisms (the file transfer protocol was primarily used), as compared to this thesis work, it can be observed that nearly the same performance differences exist between TCP and a PEP-based solution. This indicates that a level of model validity exists concerning this research. In addition, the window scale modification results were similar to those found in (Allman, 1997), with the SACK results somewhat consistent with those found in (Allman, Hayes, Kruse, and Ostermann, 1997). The previous SACK experimentation showed improvement, as compared to TCP, throughout the progression of levels of BER; the experimentation work conducted in this study showed limitations in performance benefit at BERs lower than 10^{-6}. The prior research did, however, make use of an actual satellite transmission system and a modeling environment that was closely matched to the satellite transmission system's performance characteristics; this allowed a level of realism (such as error bursts, which were not encountered in this thesis work due to software error simulation limitations) not attained in this research and likely resulted in the performance result differences.

Limitations

It is of course necessary to restate the limitations identified prior to execution of this research and those discovered during the experimentation process. First and foremost, it is obvious that use of modeling does not capture all of the idiosyncrasies, such as weather effects, solar disturbances, etc., of a live satellite transmission network (and associated network traffic); though use of such a transmission system would have been preferred, access to military satellite resources would have required a considerable amount of flexibility with respect to time and availability due to on-going military operations and low level of precedence afforded to research over military spacecraft. In addition, not all of the equipment (crypto gear, modems, multiplexers, etc.) used in conjunction with a satellite transmission were represented; the additional data processing time required for these devices may result in higher levels of latency, making the results of this experimentation somewhat inflated. It is believed, however, that the processing time required for these middle boxes would be a minor addition to the latency encountered on the wireless transmission link. Though the experimental model presented in this study may not be completely accurate, it does present findings and a methodology that can be built upon to further robust future research endeavors.

Future Research Recommendations

There are several areas for improvement of this study that could lead to additional areas of study. The most notable of these recommendations is the use of an actual military satellite communication system in tandem with a live network that makes use of a variety of network traffic that is representative of daily military operations. This type of experimentation would capture a more thorough understanding of the performance of TCP enhancement mechanisms with respect to periods of error bursts on the satellite link, dynamic traffic loads, and the effect of middle boxes (such as multiplexers, fiber optic modems, etc.) have on data stream processing, in particular the time required to process and its effect on RTT calculations made by TCP and associated enhancement techniques. In lieu of access to an actual military communication system, it is recommended that other types of traffic flows (such as those that use the file transfer protocol and the hyper text transfer protocol) be conducted over a simulated network in order to gain insight to the performance gains experienced due to use of a TCP enhancement technique. Finally, use of multiple clients should also be studied in order to ascertain the effect that multiple TCP pipes have when used with a TCP modification.

Chapter Summary

The various methods of TCP modification demonstrated varying levels of performance improvement with respect to a standard TCP implementation. The study results determined that a PEP-based solution, that makes use of the SCPS-TP standard, provided the greatest gains in terms of throughput performance. The standard improvement mechanisms of window scale and window scale/SACK also provided significant improvement, but at a level much lower than SCPS-TP. Finally, the SACK technique provided greater performance, as compared to TCP, at high levels of BER, but did not perform in a manner that was significantly different from TCP at lower error rates. From these results, it was possible to construct a decision inequality to aid satellite acquisition and planning personnel in the determination of whether or not to implement a TCP modification; study findings will also aid in the selection of a particular TCP modification technique.

Bibliography

Abdelmoumen, Raja, Mahommad Malli, and Chadi Barakat. "Analysis of TCP Latency over Wireless Links Supporting FEC/ARQ-SR for Error Recovery" IEEE International Conference on Communications 27(1): 3994-3998 (June 2004).

Allman, et al. *Request for Comments 2488: Enhancing TCP Over Satellite Channels Using Standard Mechanisms*. 1-19. 1999.

Allman, Mark. *Improving TCP Performance Over Satellite Channels*. MS thesis, Ohio University, 1997.

Allman, Mark, Chris Hayes, Hans Kruse, and Shawn Ostermann. "TCP Performance Over Satellite Links." *Proceedings of the 5th International Conference on Telecommunication Systems*. 1-14. March 1997.

Allman, Mark, Vern Paxson, and W. Richard Stevens. *Request for Comments 2581: TCP Congestion Control*. 1-14. 1999.

Balakrishnan, Hari, Venkata N. Padmanabhan, and Randy H. Katz. "The Effects of Asymmetry on TCP Performance." *Mobile Networks and Applications*, 4(3): 219-241 (1999).

Blum, Richard. *Network Performance Open Source Toolkit Using Netperf, tcptrace, NIST Net and SSFNet*. Indianapolis: Wiley Publishing, Inc., 2003.

Border et al. *Request for Comments 3135: Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations*. 1-45. June 2001.

Broyles, Ren H. *Performance Analysis of TCP Enhancements in Satellite Data Networks*. MS thesis, AFIT/GCE/ENG/99M-02. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 1999 (ADA361633).

Brakmo, Lawrence S. and Larry L. Peterson. "TCP Vegas: End to End Congestion Avoidance on a Global Internet." *IEEE Journal on Selected Areas in Communication*, 13(8): 1465-1480 (October 1995).

Caceres, Ramon and Liviu Iftode. "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environment." *IEEE Journal on Selected Areas in Communications*, 13(5): 850-859 (June 1995).

Carroll, Stephanie E. *Improving TCP Performance by Estimating Errors in a Long Delay, High Error Rate Environment*. MS thesis, AFIT/GCS/ENG/04-04. School of Engineering and Management, Air Force Institute of Technology, Wright-Patterson AFB OH, July 2004 (ADA426634).

Dykewicz, Paul. "U.S. Military Remains Big Focus of Satellite Operators." C4ISRJournal.com (16 May 2005). 20 March 2006 http://www.c4isrjournal.com/story.php?F=852484.

Durst, Robert C., Gregory J. Miller, and Eric J. Travis. "TCP Extensions for Space Communications." *Wireless Networks*, 3(5): 389-403 (October 1997).

Ehsan, Navid, Mingyan Liu, and Roderick Ragland. "Evaluation of Performance Enhancing Proxies in Internet over Satellite." *International Journal of Communication Systems*, 16(6): 513-534 (May 2003).

Feit, Sidnie. *TCP/IP: Architecture, Protocols, and Implementation*. New York City: McGraw-Hill, Inc., 1993.

Floyd, Sally and Kevin Fall. "Promoting the Use of End-to-End Congestion Control in the Internet." IEEE ACM Transactions on Networking, 7(4): 458-472 (1999).

Fox, Richard. *Request for Comments 1106: TCP Big Window and Nak Options*. 1-13. 1989.

Ghani, Nasir and Sudhir Dixit. "TCP/IP Enhancements for Satellite Networks." *IEEE Communications Magazine*: 64-72 (July 1999).

Held, Gilbert. *Understanding Data Communications: Sixth Edition*. Indianapolis: New Riders Publishing, 1999.

Henderson, Thomas R. and Randy H. Katz. “Transport Protocols for Internet-Compatible Satellite Networks.” *IEEE Journal on Selected Areas in Communications*, 17(2): 326-344 (February 1999).

Hoe, Janey. “Improving the Start-up Behavior of a Congestion Control Scheme for TCP.” *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*. 1-11. August 1996.

Iperf. Version 2.0.2 Computer software. University of Illinois, Urbana-Champaign, 1999-2005.

Ishac, Joseph. “Survey of Header Compression Techniques.” Technical Report TM-2001-211154 to NASA. September 2001.

Ishac, Joseph and Mark Allman. “On the Performance of TCP Spoofing in Satellite Networks.” *IEEE Milcom*: 1-5 (October 2001).

Jacobson, Van and Bob Braden. *Request for Comments 1072: TCP Extensions for Long-Delay Paths*. 1-16. October 1988.

Jacobson, Van, Bob Braden, and Dave Borman. *Request for Comments 1323: TCP Extensions for High Performance*. 1-37. May 1992.

Jacobson, Van and Michael Karels. “Congestion Avoidance and Control.” *Proceedings of SIGCOMM ‘88*. 1-25. August 1988.

JMP. Version 5.1. Computer software. SAS Institute, Inc., Cary NC, 1989-2003.

Karn, Phil and Craig Partridge. “Improving Round-Trip Estimates in Reliable Transport Protocols.” ACM SIGCOMM: 2-7 (August 1987).

Krishman, Rajesh, et al. “Explicit Transport Error Notification (ETEN) for Error-Prone Wireless and Satellite Networks.” *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 46(3): 343-362 (October 2004).

Lammle, Todd. *CCNA: Cisco Certified Network Associate Study Guide*. San Francisco: Sybex Inc., 2000.

Luglio, Michele, Roseti Cesare, and Mario Gerla. "The Impact of Efficient Flow Control and OS Features on TCP Performance over Satellite Links." *Satellite Communications Letters*: 1-9 (June 2004).

McClave, James T., P. George Benson, and Terry Sincich. *Statistics for Business and Economics: Ninth Edition*. Upper Saddle River: Pearson Prentice Hall, 2005.

Mathis, et al. *Request for Comments 2018: TCP Selective Acknowledgment Options*: 1-12 (1996).

Miller, C. Kenneth. "Data Distribution Over IP in High Error Rate Military Radio Environments." *Proceedings of SIGCOMM '98*. 1-5. September 1998.

Mitre Corporation. "Spanner: A Software Link Emulator." Briefing Slides. January 2006.

Narasimhan, Priya, Hans Kruse, Shawn Ostermann, and Mark Allman. "On the Impact of BER on Realistic TCP Traffic in Satellite Networks." Technical Report 04-005 to International Computer Science Institute. November 2004.

Padhye, Jitendra, Victor Firoiu, Don Towsley, and Jim Kurose. "Modeling TCP Throughput: A Simple Model and its Empirical Validation." *Proceedings of SIGCOMM '98*. 1-23. September 1998.

Postel, Jon. *Request for Comments 793: Transmission Control Protocol*: 1-85 (1981).

Roddy, Dennis. *Satellite Communications: Third Edition*. New York: McGraw-Hill Companies, Inc., 2001.

Schwab, Donald P. *Research Methods for Organizational Studies: Second Edition*. Mahway, New Jersey: Lawrence Erlbaum Associates, Publishers, 2005.

Shaughnessy, Tom and Toby Velte. *CISCO: A Beginner's Guide*. Berkeley: McGraw-Hill Companies, Inc., 2000.

Spanner. Version 2. Computer software. Mitre Corporation, McLean VA, 2006.

White, Edward. Class Lecture, STAT 535, Applied Statistics for Managers II. Department of Mathematics & Statistics, Air Force Institute of Technology, Wright-Patterson AFB OH, January 21, 2005.

XipLink Gateway User Manual: Gateway Release 2.4. Product Manual. Montreal: Xiphos Technologies, Inc., 2005.

Appendix A: Xiplink Mini-Gateway Configuration

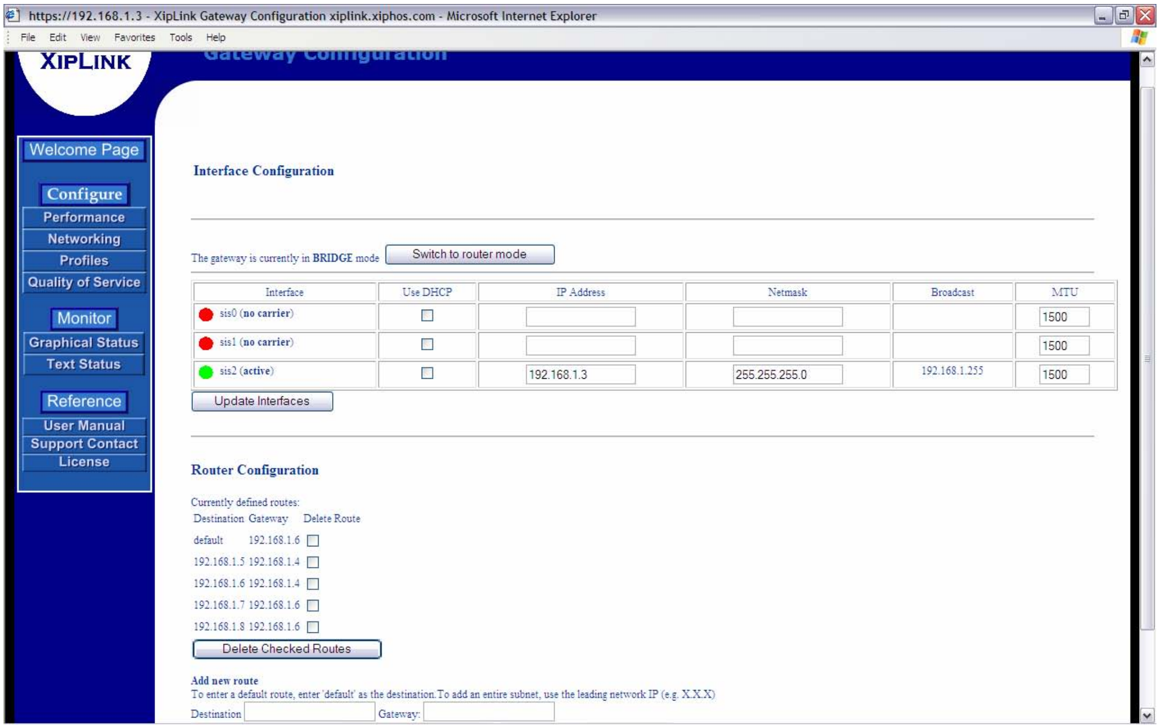

Figure 32. *SCPS10* Network Configuration

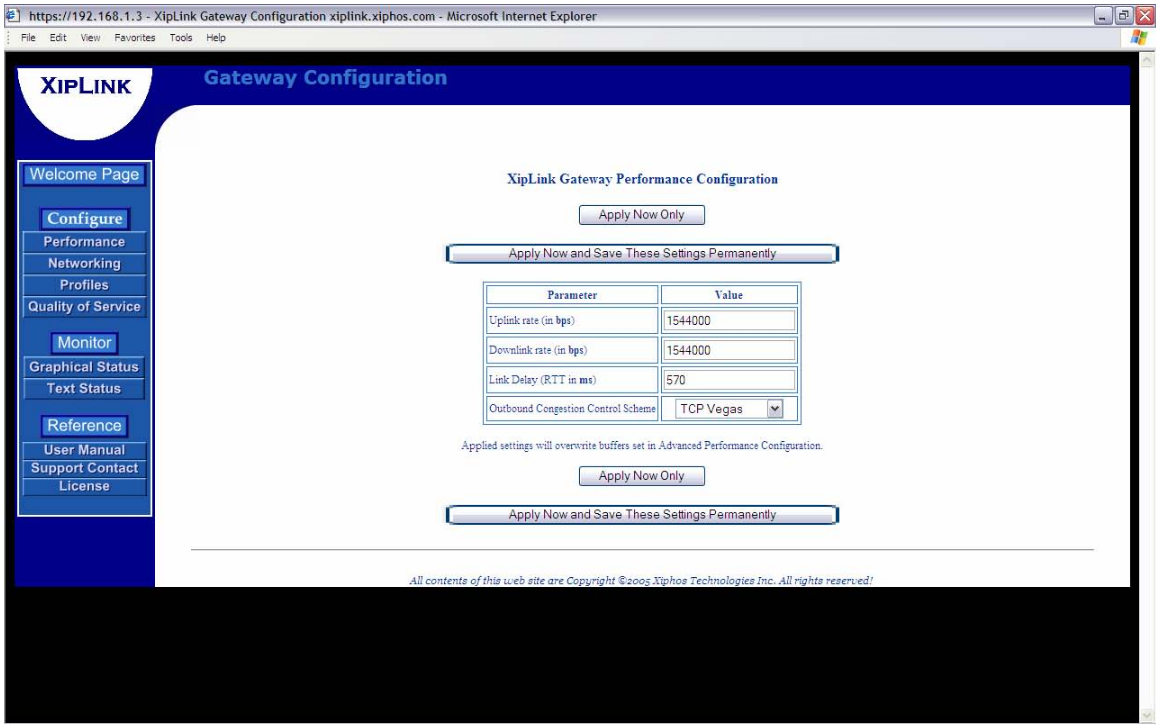

Figure 33. *SCPS10* Basic Performance Configuration

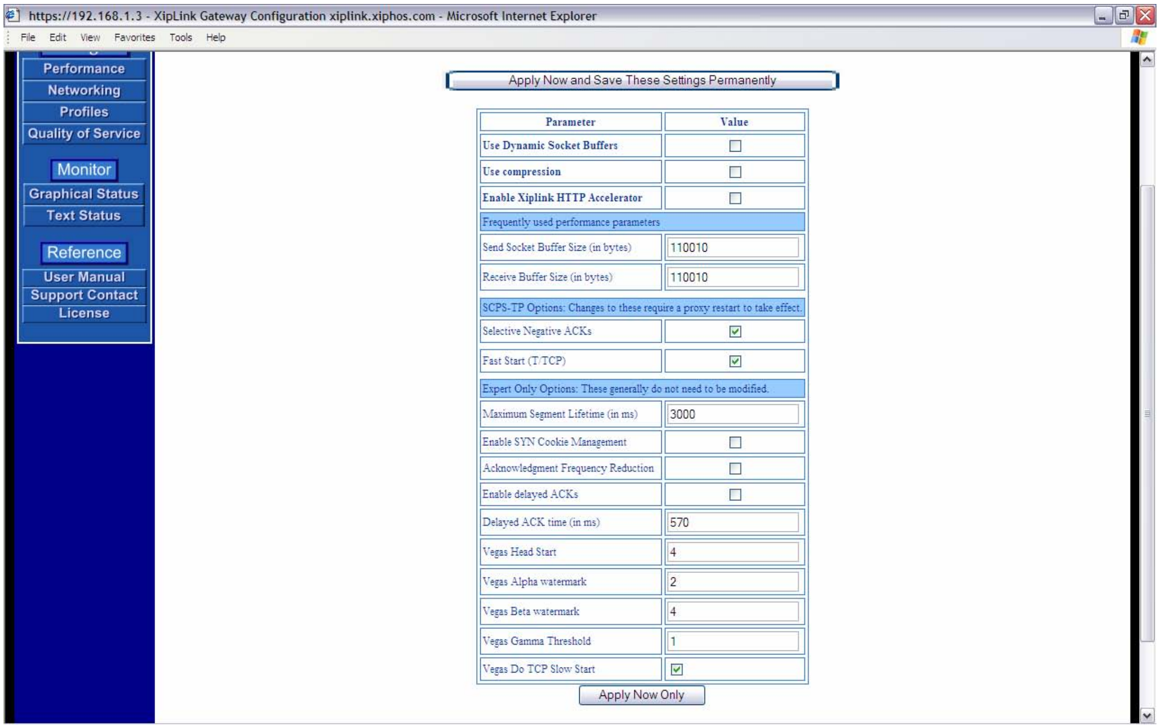

Figure 34. *SCPS10* Advance Performance Configuration

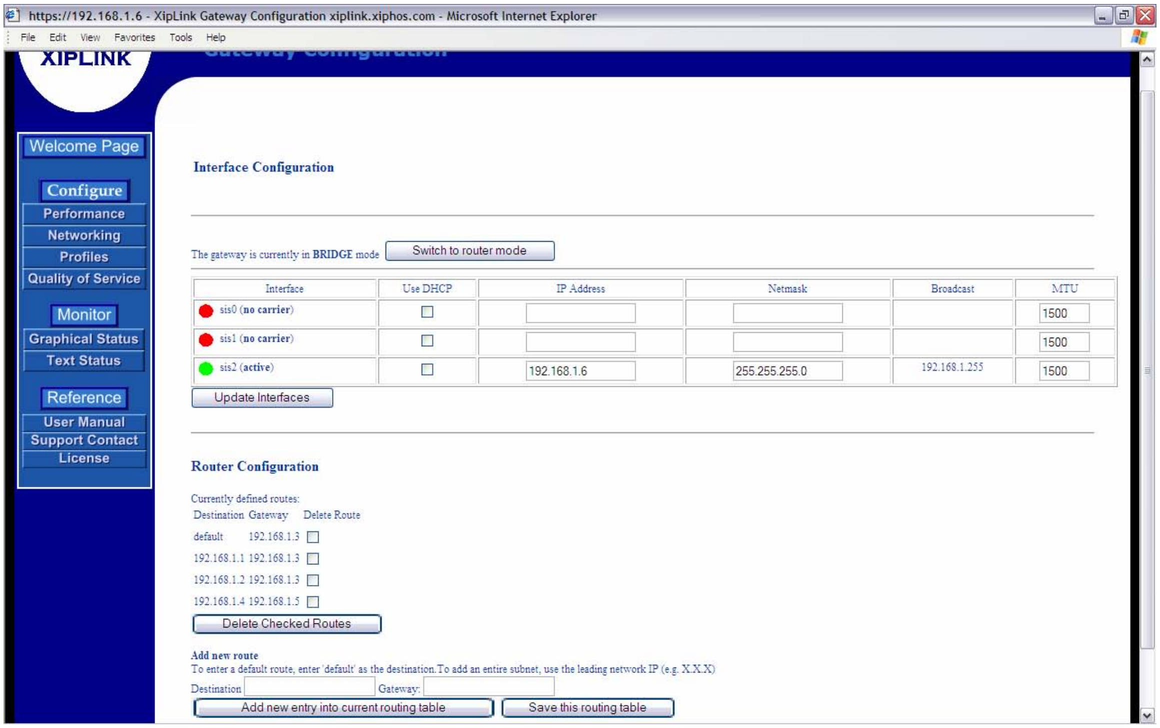

Figure 35. *SCPS20* Network Configuration

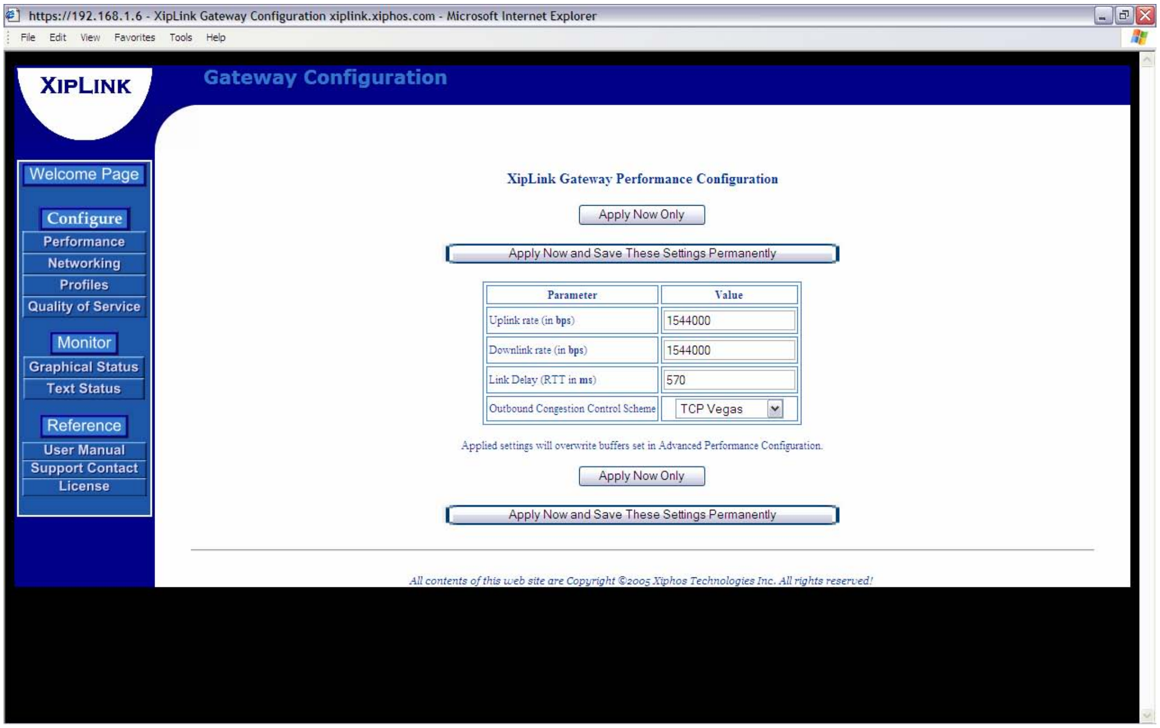

Figure 36. *SCPS20* Basic Performance Configuration

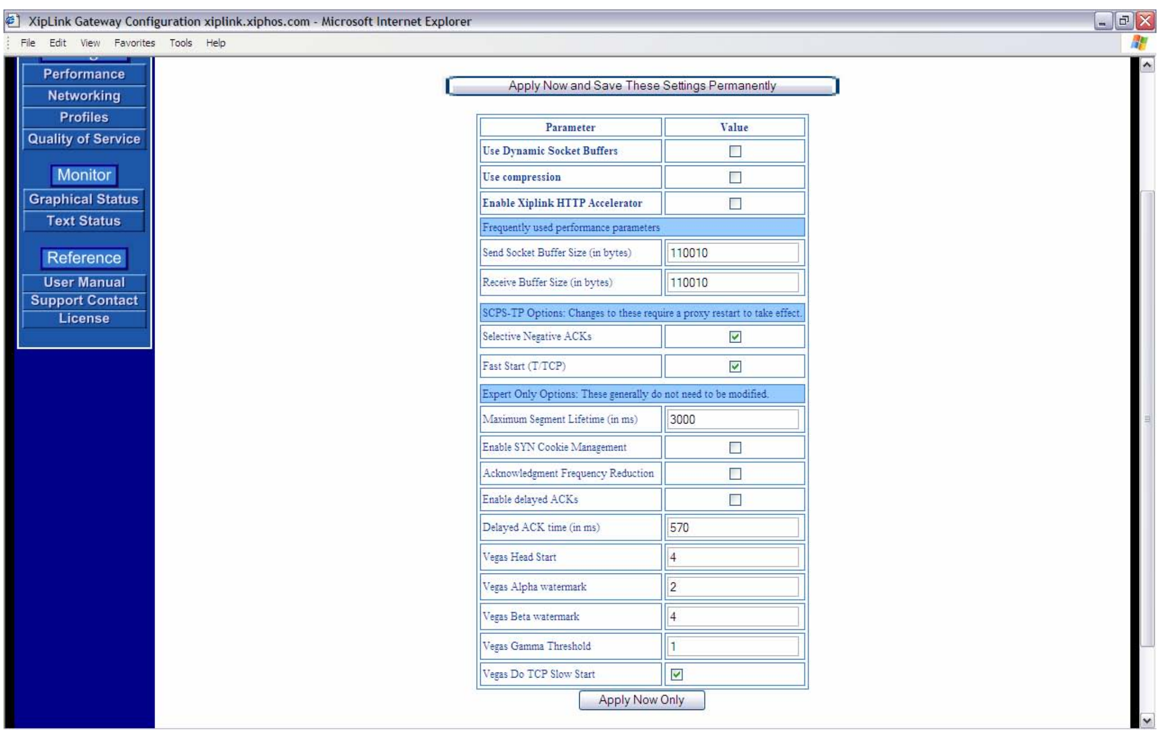

Figure 37. *SCPS20* Advance Performance Configuration

Appendix B: Data Collection Scripts

The automation of the data collection for this thesis was made possible via use of a few simple scripts. The primary script, CollectIt, was initiated on the server *spanner* and would establish the parameters (via execution of commands through the command line interface) for the spanner program. In addition, the CollectIt script would also initiate SSH sessions with server *netperfc* in order to execute scripts (known as DoE5, DoE6, DoE7, DoE8, and DoE9) found on this remote server. The scripts on *netperfc* executed Iperf commands that generated the TCP flows needed to gather the throughput data collected for each of the test scenarios. The following sections provide the syntax for each of the scripts utilized during this experimentation. Please note that the ClientDone file mentioned in each of the scripts was a dummy file passed to indicate the completion of a testing scenario; the file contained no syntax of interest.

CollectIt Script

```
#!/bin/bash

# Define functions used in this script

kill_span () {
kill -9 `ps | grep spanner | colrm 6 144`

# kill bridges

/sbin/ifconfig aif down
/usr/sbin/brctl delbr aif
/sbin/ifconfig bif down
/usr/sbin/brctl delbr bif

# restore routes

/sbin/route add -net 192.168.1.0 netmask 255.255.255.0 dev eth0
/sbin/route add -host 192.168.1.1 gw 192.168.1.3 dev eth0
```

```
}

# Start main script

echo Starting Data Collection at `date`

# Set neg 5 error rate

echo Starting NEG 5 at `date`

# Make sure there are no running spanner instances

kill_span

# Make sure file flag erased

/bin/rm -f /root/ClientDone

# Run spanner in background with neg5

/usr/local/src/SPANNER_II/spanner -I eth0 -R eth1 -d 0.285 -i 1544000 -r 1544000 -e
0.00001 &

# wait for spanner to start working properly

sleep 10s
route add -net 192.168.1.0 netmask 255.255.255.0 dev aif
route add -host 192.168.1.1 gw 192.168.1.3
sleep 10s

# start remote data collection

ssh netperfc /root/DoE5

# wait until netperfc is done

until test -f /root/ClientDone
do
	sleep 5m
done

# kill spanner

kill_span

# let things settle
```

```
sleep 5
echo NEG 5 done at `date`

# repeat

# Set neg 6 error rate

echo Starting NEG 6 at `date`

# Make sure there are no running spanner instances

kill_span

# Make sure file flag erased

/bin/rm -f /root/ClientDone

# Run spanner in background with neg6

/usr/local/src/SPANNER_II/spanner -I eth0 -R eth1 -d 0.285 -i 1544000 -r 1544000 -e 0.000001 &

# wait for spanner to start working properly

sleep 10s
route add -net 192.168.1.0 netmask 255.255.255.0 dev aif
route add -host 192.168.1.1 gw 192.168.1.3
sleep 10s

# start remote data collection

ssh netperfc /root/DoE6

# wait until netperfc is done

until test -f /root/ClientDone
do
	sleep 5m
done

# kill spanner

kill_span

# let things settle
```

```
sleep 5
echo NEG 6 done at `date`

# repeat

# Set neg 7 error rate

echo Starting NEG 7 at `date`

# Make sure there are no running spanner instances

kill_span

# Make sure file flag erased

/bin/rm -f /root/ClientDone

# Run spanner in background with neg7

/usr/local/src/SPANNER_II/spanner -I eth0 -R eth1 -d 0.285 -i 1544000 -r 1544000 -e
0.0000001 &

# wait for spanner to start working properly

sleep 10s
route add -net 192.168.1.0 netmask 255.255.255.0 dev aif
route add -host 192.168.1.1 gw 192.168.1.3
sleep 10s

# start remote data collection

ssh netperfc /root/DoE7

# wait until netperfc is done

until test -f /root/ClientDone
do
	sleep 5m
done

# kill spanner

kill_span

# let things settle
```

```
sleep 5
echo NEG 7 done at `date`

# repeat

# Set neg 8 error rate

echo Starting NEG 8 at `date`

# Make sure there are no running spanner instances

kill_span

# Make sure file flag erased

/bin/rm -f /root/ClientDone

# Run spanner in background with neg8

/usr/local/src/SPANNER_II/spanner -I eth0 -R eth1 -d 0.285 -i 1544000 -r 1544000 -e
0.00000001 &
# wait for spanner to start working properly
sleep 10s
route add -net 192.168.1.0 netmask 255.255.255.0 dev aif
route add -host 192.168.1.1 gw 192.168.1.3
sleep 10s

# start remote data collection

ssh netperfc /root/DoE8

# wait until netperfc is done

until test -f /root/ClientDone
do
	sleep 5m
done

# kill spanner

kill_span

# let things settle

sleep 5
```

```
echo NEG 8 done at `date`

# repeat

# Set neg 9 error rate

echo Starting NEG 9 at `date`

# Make sure there are no running spanner instances

kill_span

# Make sure file flag erased

/bin/rm -f /root/ClientDone

# Run spanner in background with neg9

/usr/local/src/SPANNER_II/spanner -I eth0 -R eth1 -d 0.285 -i 1544000 -r 1544000 -e
0.000000001 &

# wait for spanner to start working properly

sleep 10s
route add -net 192.168.1.0 netmask 255.255.255.0 dev aif
route add -host 192.168.1.1 gw 192.168.1.3
sleep 10s

# start remote data collection

ssh netperfc /root/DoE9

# wait until netperfc is done

until test -f /root/ClientDone
do
	sleep 5m
done

# kill spanner

kill_span

# let things settle

sleep 5
```

```
echo NEG 9 done at `date`

# repeat

echo Done at `date`
```

DoE5 Script

```
# Script for executing E5 and 128k packet testing

echo "E5 and 128K packet testing began at `date`" > Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
iperf -c 192.168.1.8 -t 300 -f k -l 128k >> Results128kE5
echo "E5 and 128K packet testing completed at `date`" >> Results128kE5

# Script for executing E5 and 512k packet testing

echo "E5 and 512K packet testing began at `date`" > Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
```

```
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
iperf -c 192.168.1.8 -t 300 -f k -l 512k >> Results512kE5
echo "E5 and 512K packet testing completed at `date`" >> Results512kE5
scp /root/ClientDone root@192.168.1.4:/root/ClientDone
```

DoE6, DoE7, DoE8, and DoE9 Scripts

The scripts for the DeE6, DoE7, DoE8, and DoE9 scripts are nearly identical to that found in the DoE5 script; the only difference is the naming of the results file. The results files were named so that the error rate utilized was the visual discriminator.

Appendix C: Research Data Set

Table 31. TCP (No Enhancement) Data Set for 128kb Packet Scenarios

128kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	**1.0E-05**	**1.0E-06**	**1.0E-07**	**1.0E-08**	**1.0E-09**
TCP (No Enhancement)	39.5	186	387	418	430
TCP (No Enhancement)	40.9	196	360	430	433
TCP (No Enhancement)	44.1	183	339	414	433
TCP (No Enhancement)	40.2	194	346	427	433
TCP (No Enhancement)	21.1	159	365	429	430
TCP (No Enhancement)	44.8	178	349	426	433
TCP (No Enhancement)	38.4	190	344	418	430
TCP (No Enhancement)	40.8	200	341	428	433
TCP (No Enhancement)	42.0	168	373	420	430
TCP (No Enhancement)	39.3	177	341	428	433
TCP (No Enhancement)	40.1	186	372	422	430
TCP (No Enhancement)	42.2	176	336	428	433
TCP (No Enhancement)	40.2	177	389	414	430
TCP (No Enhancement)	41.9	171	340	430	433
TCP (No Enhancement)	42.8	188	357	420	433
TCP (No Enhancement)	36.4	154	357	421	433
TCP (No Enhancement)	43.0	164	373	421	430
TCP (No Enhancement)	41.3	184	375	423	433
TCP (No Enhancement)	39.5	191	390	418	430
TCP (No Enhancement)	37.6	178	383	425	430
TCP (No Enhancement)	44.2	163	358	417	430
TCP (No Enhancement)	43.2	167	359	430	433
TCP (No Enhancement)	43.4	157	373	430	430
TCP (No Enhancement)	44.3	180	343	427	433
TCP (No Enhancement)	43.0	182	371	428	433
TCP (No Enhancement)	42.0	176	355	421	433
TCP (No Enhancement)	40.6	167	340	434	430
TCP (No Enhancement)	40.4	197	345	425	433
TCP (No Enhancement)	42.0	171	374	423	430
TCP (No Enhancement)	40.1	177	336	426	433

Table 32. TCP (No Enhancement) Data Set for 512kb Packet Scenarios

512kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	**1.0E-05**	**1.0E-06**	**1.0E-07**	**1.0E-08**	**1.0E-09**
TCP (No Enhancement)	40.4	183	343	430	430
TCP (No Enhancement)	40	180	358	429	434
TCP (No Enhancement)	42.7	174	382	424	433
TCP (No Enhancement)	40.5	177	349	419	434
TCP (No Enhancement)	43.6	182	372	429	430
TCP (No Enhancement)	42.6	191	335	426	430
TCP (No Enhancement)	39.4	171	376	426	434
TCP (No Enhancement)	39.1	176	348	419	430
TCP (No Enhancement)	39.1	181	346	426	434
TCP (No Enhancement)	38.4	159	323	415	433
TCP (No Enhancement)	38.4	185	338	422	434
TCP (No Enhancement)	42.2	158	381	426	429
TCP (No Enhancement)	39.7	184	369	430	434
TCP (No Enhancement)	44.7	172	361	425	430
TCP (No Enhancement)	41.3	178	347	430	434
TCP (No Enhancement)	41.2	167	369	420	429
TCP (No Enhancement)	42.5	194	343	429	434
TCP (No Enhancement)	39.8	159	384	424	429
TCP (No Enhancement)	42.8	182	373	425	434
TCP (No Enhancement)	44.7	164	385	430	429
TCP (No Enhancement)	42.1	169	344	427	434
TCP (No Enhancement)	39.2	167	358	430	430
TCP (No Enhancement)	42	168	337	426	434
TCP (No Enhancement)	42.9	165	370	421	430
TCP (No Enhancement)	39.4	169	323	426	429
TCP (No Enhancement)	40.6	171	343	427	434
TCP (No Enhancement)	41	172	379	430	430
TCP (No Enhancement)	40.6	173	355	422	434
TCP (No Enhancement)	42.5	174	371	430	430
TCP (No Enhancement)	38.5	197	363	429	429

Table 33. SACK Data Set for 128kb Packet Scenarios

128kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	1.0E-05	1.0E-06	1.0E-07	1.0E-08	1.0E-09
SACK	45.8	189	366	420	430
SACK	48.3	169	391	424	429
SACK	51.2	176	383	418	433
SACK	49.6	201	398	425	433
SACK	48.9	182	389	417	433
SACK	49.3	186	372	425	430
SACK	45.4	178	379	417	433
SACK	49.2	158	391	419	430
SACK	46.5	181	373	425	431
SACK	54.3	171	393	423	430
SACK	48.6	192	380	423	431
SACK	50.4	195	387	426	429
SACK	48.0	170	391	418	431
SACK	48.6	172	372	427	429
SACK	49.6	181	375	419	433
SACK	54.1	175	389	421	430
SACK	43.4	188	394	421	430
SACK	48.2	190	383	428	430
SACK	46.9	204	379	419	431
SACK	52.5	171	384	419	430
SACK	48.4	176	371	426	430
SACK	48.8	175	368	419	430
SACK	51.4	188	394	419	430
SACK	46.3	172	371	423	430
SACK	47.5	193	378	417	430
SACK	53.6	165	397	420	433
SACK	44.3	179	392	424	430
SACK	51.8	173	378	419	433
SACK	46.5	191	367	424	429
SACK	53.6	177	389	422	433

Table 34. SACK Data Set for 512kb Packet Scenarios

512kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	1.0E-05	1.0E-06	1.0E-07	1.0E-08	1.0E-09
SACK	47.6	163	373	419	430
SACK	51.1	183	390	420	434
SACK	49.7	173	365	428	430
SACK	47.2	199	368	416	434
SACK	49.3	165	390	426	430
SACK	49.8	192	382	422	434
SACK	49.0	177	384	424	430
SACK	50.8	181	370	423	434
SACK	49.3	182	389	421	429
SACK	50.2	189	373	418	434
SACK	47.0	190	370	423	429
SACK	48.1	184	382	419	434
SACK	49.3	184	371	423	430
SACK	46.2	178	379	420	434
SACK	46.3	184	386	423	430
SACK	54.7	182	376	428	434
SACK	49.3	176	383	427	430
SACK	51.6	182	378	419	434
SACK	49.6	188	373	423	430
SACK	45.7	191	384	423	434
SACK	51.9	177	388	423	430
SACK	51.5	191	376	420	434
SACK	49.7	190	387	419	433
SACK	46.9	178	390	420	434
SACK	49.0	163	382	427	434
SACK	51.0	179	390	420	432
SACK	47.2	174	385	419	434
SACK	45.6	181	389	430	430
SACK	49.5	187	376	426	434
SACK	46.8	215	366	428	430

Table 35. SCPS-TP Data Set for 128kb Packet Scenarios

128kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	1.0E-05	1.0E-06	1.0E-07	1.0E-08	1.0E-09
SCPS-TP	167.0	290	248	1421	545
SCPS-TP	124.0	427	1208	1422	1446
SCPS-TP	151.0	424	1183	1434	1444
SCPS-TP	170.0	454	1201	1431	1443
SCPS-TP	164.0	417	1180	1446	1444
SCPS-TP	184.0	429	1188	1425	1445
SCPS-TP	147.0	457	1223	1422	1444
SCPS-TP	155.0	463	1184	1433	1444
SCPS-TP	151.0	458	1225	1410	1445
SCPS-TP	127.0	404	1232	1431	1444
SCPS-TP	141.0	423	1181	1420	1444
SCPS-TP	143.0	453	1229	1434	1443
SCPS-TP	157.0	409	1238	1421	1446
SCPS-TP	198.0	380	1143	1434	1445
SCPS-TP	167.0	248	1196	1426	1444
SCPS-TP	183.0	475	1151	1441	1444
SCPS-TP	158.0	428	1199	1420	1443
SCPS-TP	170.0	473	1140	1430	1445
SCPS-TP	166.0	408	1183	1434	1445
SCPS-TP	96.1	387	1145	1439	1444
SCPS-TP	195.0	437	1186	1415	1445
SCPS-TP	148.0	394	1219	1426	1442
SCPS-TP	163.0	415	1214	1419	1443
SCPS-TP	91.0	430	1182	1426	1446
SCPS-TP	109.0	403	1157	1432	1443
SCPS-TP	89.1	424	1219	1423	1444
SCPS-TP	146.0	405	1163	1414	1444
SCPS-TP	169.0	247	1147	1425	1441
SCPS-TP	146.0	413	1194	1416	1443
SCPS-TP	145.0	174	1178	1413	1444

Table 36. SCPS-TP Data Set for 512kb Packet Scenarios

512kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	**1.0E-05**	**1.0E-06**	**1.0E-07**	**1.0E-08**	**1.0E-09**
SCPS-TP	166.0	401	1188	1427	1444
SCPS-TP	156.0	410	1234	1423	1445
SCPS-TP	145.0	394	1237	1432	1376
SCPS-TP	117.0	442	1130	1408	1447
SCPS-TP	165.0	441	1166	1432	1443
SCPS-TP	171.0	344	1155	1430	1444
SCPS-TP	163.0	430	1171	1433	1442
SCPS-TP	151.0	230	1202	1420	1444
SCPS-TP	127.0	400	1173	1435	1443
SCPS-TP	174.0	264	1193	1417	1447
SCPS-TP	164.0	293	1140	1415	1444
SCPS-TP	148.0	436	1214	1435	1445
SCPS-TP	158.0	412	1211	1428	1446
SCPS-TP	168.0	403	1202	1403	1444
SCPS-TP	141.0	463	1155	1435	1447
SCPS-TP	91.2	414	1175	1420	1441
SCPS-TP	139.0	452	1202	1422	1445
SCPS-TP	124.0	415	1201	1418	1445
SCPS-TP	174.0	405	1207	1415	1445
SCPS-TP	138.0	432	1123	1418	1445
SCPS-TP	157.0	425	1217	1432	1440
SCPS-TP	157.0	405	1203	1427	1445
SCPS-TP	190.0	426	1113	1421	1443
SCPS-TP	154.0	448	1199	1429	1444
SCPS-TP	121.0	492	1214	1438	1445
SCPS-TP	164.0	445	1174	1426	1447
SCPS-TP	149.0	483	1125	1446	1443
SCPS-TP	161.0	429	1158	1420	1443
SCPS-TP	130.0	379	1226	1432	1445
SCPS-TP	147.0	140	285	1427	1444

Table 37. Window Scale Data Set for 128kb Packet Scenarios

128kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	**1.0E-05**	**1.0E-06**	**1.0E-07**	**1.0E-08**	**1.0E-09**
Window Scale	42.6	190	558	901	902
Window Scale	42.5	202	609	901	901
Window Scale	40.8	197	528	901	901
Window Scale	43.7	188	601	901	902
Window Scale	40.3	187	472	901	901
Window Scale	42.4	198	551	901	901
Window Scale	38.5	168	445	901	901
Window Scale	43.5	218	592	901	901
Window Scale	44.9	171	486	901	901
Window Scale	40.8	208	490	901	902
Window Scale	38.8	158	557	901	901
Window Scale	41.7	173	606	901	901
Window Scale	39.3	213	564	901	901
Window Scale	42.2	187	507	902	901
Window Scale	39.5	177	483	901	901
Window Scale	44.3	205	620	901	901
Window Scale	45.0	200	432	901	901
Window Scale	42.9	181	600	902	901
Window Scale	39.0	179	567	901	901
Window Scale	44.4	203	526	901	901
Window Scale	44.7	190	595	901	901
Window Scale	42.9	179	434	901	901
Window Scale	39.9	189	504	901	901
Window Scale	42.5	197	595	901	901
Window Scale	43.8	167	556	901	901
Window Scale	40.8	195	588	901	901
Window Scale	41.8	189	528	901	901
Window Scale	42.7	189	568	901	901
Window Scale	37.8	191	519	901	901
Window Scale	39.6	178	488	901	901

Table 38. Window Scale Data Set for 512kb Packet Scenarios

512kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	1.0E-05	1.0E-06	1.0E-07	1.0E-08	1.0E-09
Window Scale	40.2	184	515	904	904
Window Scale	41.2	178	449	904	904
Window Scale	41.0	151	480	904	904
Window Scale	42.0	163	516	904	904
Window Scale	43.6	172	609	904	904
Window Scale	39.6	167	515	904	904
Window Scale	41.5	164	652	903	904
Window Scale	44.6	211	458	904	904
Window Scale	42.1	195	516	904	904
Window Scale	43.6	174	480	904	904
Window Scale	42.6	192	555	904	904
Window Scale	41.9	206	581	904	904
Window Scale	44.1	156	639	904	904
Window Scale	40.3	163	536	904	904
Window Scale	43.3	184	547	904	904
Window Scale	43.8	176	527	904	904
Window Scale	41.8	177	566	904	904
Window Scale	39.3	164	505	904	904
Window Scale	41.2	174	576	904	904
Window Scale	46.1	178	515	904	904
Window Scale	39.9	180	538	904	904
Window Scale	40.4	177	491	904	904
Window Scale	40.8	212	639	904	904
Window Scale	40.4	174	488	904	904
Window Scale	40.5	199	506	904	904
Window Scale	39.9	195	560	904	904
Window Scale	42.2	197	576	904	904
Window Scale	44.7	194	501	904	904
Window Scale	41.2	187	516	904	904
Window Scale	46.4	197	556	864	904

Table 39. Window Scale/SACK Data Set for 128kb Packet Scenarios

128kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	**1.0E-05**	**1.0E-06**	**1.0E-07**	**1.0E-08**	**1.0E-09**
Window Scale/SACK	48.3	191	660	833	901
Window Scale/SACK	48.3	181	664	835	902
Window Scale/SACK	46.4	174	570	850	902
Window Scale/SACK	47.6	154	600	799	901
Window Scale/SACK	50.4	203	564	810	901
Window Scale/SACK	48.9	174	653	832	901
Window Scale/SACK	51.2	171	607	830	901
Window Scale/SACK	51.2	184	669	849	901
Window Scale/SACK	49.2	189	596	828	901
Window Scale/SACK	49.7	169	661	818	901
Window Scale/SACK	49.4	180	570	816	901
Window Scale/SACK	52.3	166	674	815	901
Window Scale/SACK	47.7	178	629	829	901
Window Scale/SACK	55.7	169	644	846	901
Window Scale/SACK	47.0	194	631	845	901
Window Scale/SACK	44.6	157	589	840	901
Window Scale/SACK	49.5	204	631	820	901
Window Scale/SACK	47.9	173	589	782	901
Window Scale/SACK	47.5	206	664	835	901
Window Scale/SACK	49.8	186	568	860	901
Window Scale/SACK	52.0	178	685	819	901
Window Scale/SACK	51.7	175	622	838	901
Window Scale/SACK	49.6	183	640	830	901
Window Scale/SACK	50.5	172	592	844	901
Window Scale/SACK	52.3	198	633	873	901
Window Scale/SACK	49.4	179	574	834	901
Window Scale/SACK	53.6	213	630	812	901
Window Scale/SACK	50.9	173	597	848	901
Window Scale/SACK	44.1	195	632	815	901
Window Scale/SACK	48.1	162	593	817	901

Table 40. Window Scale/SACK Data Set for 512kb Packet Scenarios

512kb Packet Scenario	Throughput (kbps) Based on BER				
Enhancement Technique	**1.0E-05**	**1.0E-06**	**1.0E-07**	**1.0E-08**	**1.0E-09**
Window Scale/SACK	46.9	181	578	822	904
Window Scale/SACK	47.2	185	566	849	904
Window Scale/SACK	52.7	181	620	791	904
Window Scale/SACK	49.8	170	584	824	904
Window Scale/SACK	45.1	165	599	863	904
Window Scale/SACK	49.3	193	601	849	904
Window Scale/SACK	46.9	195	622	861	904
Window Scale/SACK	51.6	188	590	855	904
Window Scale/SACK	51.6	197	577	824	904
Window Scale/SACK	47.2	176	623	854	904
Window Scale/SACK	56.0	226	545	831	904
Window Scale/SACK	48.7	208	607	823	904
Window Scale/SACK	50.4	199	618	825	904
Window Scale/SACK	47.7	163	669	820	904
Window Scale/SACK	44.8	196	645	845	904
Window Scale/SACK	50.4	191	562	839	904
Window Scale/SACK	48.0	192	634	805	904
Window Scale/SACK	46.8	179	586	844	904
Window Scale/SACK	54.1	186	608	833	904
Window Scale/SACK	50.3	184	567	841	904
Window Scale/SACK	48.8	175	611	856	904
Window Scale/SACK	50.2	213	628	836	904
Window Scale/SACK	45.6	206	623	837	904
Window Scale/SACK	53.2	190	593	849	904
Window Scale/SACK	45.0	181	564	845	904
Window Scale/SACK	47.5	196	630	844	904
Window Scale/SACK	50.4	191	622	817	904
Window Scale/SACK	47.2	197	640	838	904
Window Scale/SACK	48.6	169	593	856	904
Window Scale/SACK	48.5	214	636	771	904

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
23-03-2006	**Master's Thesis**	Apr 2005 – Apr 2006

4. TITLE AND SUBTITLE A Comparative Analysis of Transmission Control Protocol Improvement Techniques Over Space-Based Transmission Media	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Lawson, Joseph M., Captain, USAF	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GIR/ENV/06M-08
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSOR/MONITOR'S ACRONYM(S)**
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The purpose of this study was to assess the throughput improvement afforded by the various TCP optimization techniques, with respect to a simulated geosynchronous satellite system, in order to provide a cost justification for the implementation of a given enhancement technique. The research questions were answered through model and simulation of a satellite transmission system via a Linux-based network topology; results of the simulation were analyzed primarily via a non-parametric method to ascertain performance differences between the various TCP optimization techniques. It was determined that each technique studied, which included the Space Communication Protocol Standard – Transport Protocol (SCPS-TP), window scale, selective acknowledgements (SACKs), and combinational use of the window scale and SACK mechanisms, provided varying levels of improvement as compared to a standard TCP implementation. In terms of throughput, SCPS-TP provided the greatest overall improvement, with window scale and window scale/SACK techniques providing significant benefits at low levels of bit error rate (BER). The SACK modification improved throughput performance at high levels of BER, but performed at levels comparable to standard TCP during scenarios with lower BER levels. These findings will be of assistance to communications planners in deciding whether or not to implement a given enhancement or deciding which technique to utilize.

15. SUBJECT TERMS
Networks, Satellite Communications, Satellite Networks, Internet, Communications Protocols, Wireless, Transmission Control Protocol, Selective Acknowledgements, Window Scale, Space Communication Protocol Standard-Transport Protocol, Performance Enhancing Proxies.

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Michael R. Grimaila, ENV
REPORT	ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	121	(937) 255-3636, ext 4800; e-mail: Michael.Grimaila@afit.edu

Standard Form 298 (Rev: 8-98)
Prescribed by ANSI Std. Z39-18