INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI[®]

Performance of Internet Applications over Wireless CDMA

By

Hamid Mahmood Syed

B.E. (Electrical Engineering),

N.E.D. University of Engineering & Technology, Karachi, Pakistan

**A thesis submitted to the Faculty of Graduate Studies and
Research in partial fulfillment of the requirements of the degree of**

Master of Engineering

**Ottawa-Carleton Institute of Electrical and Computer Engineering
Faculty of Engineering**

**Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario K1S 5B6, Canada**

June 15, 1999

© 1999, Hamid Mahmood Syed

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-48461-0

Canada

The undersigned recommend the Faculty of Graduate
Studies and Research the acceptance of the thesis

Performance of Internet Applications over Wireless CDMA

Submitted by Hamid Mahmood Syed, B.E.

In partial fulfillment of the requirements for the degree of
Master of Engineering

Professor Mike Devetsikiotis

Thesis supervisor

Professor Rafik A. Goubran

Chair, Department of Systems and Computer Engineering

Abstract

Due to rapid advances in the area of wireless communication and popularity of Internet, provision of Internet services over wireless is gaining importance. Transport Control Protocol (TCP) is a reliable, end-to-end, protocol in the Internet Protocol Suite and is widely used in protocols like file transfer, web browsing and e-mail. TCP was designed for fixed networks where channel error rates are very low and congestion is the primary cause of packet losses. However, when used in mobile environments, which are characterized by high channel errors and frequent user mobility, packet losses increase substantially. These losses severely degrade the performance TCP and the quality of Internet applications.

A great deal of research is being carried out to tune TCP against wireless losses and improve the performance of Internet applications in these environments. In this study, we provided a survey of various approaches that are suggested to improve TCP over wireless.

Wide Band Spread Spectrum Digital Cellular Standard (IS-99) has introduced a middle layer, Radio Link Protocol, over IS-95 physical layer. Radio Link Protocol operates on its ARQ mechanism to recover losses due to wireless channel errors and hides them from the transport layer. The focus of this work is to identify the performance parameters and their dependency on other network parameters. The evaluation of Internet traffic performance using the protocol stack of IS-99 has been performed. Moreover, the effects of identified network parameters on performance of Internet applications have also been examined.

Acknowledgments

All praise to Allah (God) who gave me the capability to achieve the objectives of this study and made things easier for me.

I am sincerely grateful to my thesis supervisor, Professor Mike Devetsikiotis for his guidance, encouragement, motivation and continued support in my research. His vast knowledge and experience in the area of Networking proved very useful during the research. I am also grateful to, Dr. Kaushik Das, for devoting substantial time and energy in discussions on thesis-related research issues and for contributing invaluable solutions and ideas.

I am thankful to my parents and family members for their support and understanding during the course of my studies.

My colleagues at Broadband Networks Lab have been very helpful and supportive for which I am thankful to them. At the same time, I would also like to pay my gratitude to all those who have helped me in this research.

I would like to thank Communications and Information Technology Ontario (CITO) for providing financial assistance and the opportunity of research.

Table of Contents

ABSTRACT	III
ACKNOWLEDGMENTS.....	IV
TABLE OF CONTENTS	V
LIST OF FIGURES.....	VIII
LIST OF TABLES.....	X
LIST OF ACRONYMS.....	XI
INTRODUCTION	1
1.1 MCTIVATION	1
1.2 THESIS CONTRIBUTION.....	4
1.3 THESIS ORGANIZATION.....	5
OVERVIEW OF TCP.....	7
2.1 TCP OVER WIRELINE NETWORKS	7
2.1.1 <i>Efficient Transmission</i>	7
2.1.2 <i>Flow Control</i>	8
2.2 TCP RESPONSE TO CONGESTION.....	9
2.2.1 <i>Slow Start</i>	10
2.2.2 <i>Congestion avoidance</i>	11
2.2.3 <i>Fast Retransmit</i>	12
2.2.4 <i>Fast Recovery</i>	12
2.3 WHAT IS DIFFERENT WITH A WIRELESS SYSTEM?	14
BACKGROUND WORK.....	19
3.1 IMPROVING TCP OVER WIRELESS: PROPOSALS	19

3.1.1 Split Connection Approach.....	20
3.1.1.1 Indirect TCP (I-TCP).....	20
3.1.2 Link-Layer Approach.....	21
3.1.2.1 The snoop Protocol.....	22
3.1.2.2 Radio Link Protocol for CDMA	24
3.1.3 End-to-End approach	25
3.1.3.1 Fast Retransmit Approach.....	25
3.1.3.2 Selective Acknowledgements.....	26
3.1.3.3 Modification to TCP as suggested by the University of Maryland	28
3.1.3.4 Mobile TCP	30
3.1.4 Selection among the Proposals.....	32
3.2 OVERVIEW OF THE RADIO LINK PROTOCOL	33
3.3 INTERNET PROTOCOL APPLICATIONS.....	35
3.3.1 File Transfer Protocol (FTP).....	35
3.3.2 Web Browsing (WWW).....	37
3.3.3 Electronic Mail (SMTP).....	37
3.4 TRAFFIC GENERATORS	38
3.4.1 WWW Traffic Generator	38
3.4.1.1 Traffic Model	39
3.4.2 FTP/SMTP Generator.....	42
DESIGN AND VALIDATION OF TCP/RLP TEST BED.....	43
4.2 MODELING TCP BEHAVIOR	45
4.2.1 Maximum Segment Size	46
4.2.2 Maximum Window Size.....	47
4.2.3 Window Initialization.....	47
4.2.4 Congestion Window Management	47
4.2.4.1 Normal operation	48
4.2.4.2 Timer Time Out occurs without receiving ACK	48

4.2.4.3 Three Duplicate ACKs received.....	49
4.2.5 <i>Communicating Transmitting and Receiving Ends</i>	50
4.2.6 <i>Introducing Errors</i>	51
4.2.7 <i>Statistics Collection</i>	53
4.2.7.1 TCP Throughput	53
4.2.7.2 Mean Delay	54
4.3 SIMULATOR VALIDATION	56
PERFORMANCE OF INTERNET APPLICATIONS	66
5.1 PERFORMANCE PARAMETERS	66
5.2 SIMULATION ENVIRONMENT	69
5.3 RESULTS AND ANALYSIS	71
5.3.1 <i>Performance of FTP/SMTP</i>	71
5.3.1.1 Effect of Number of RLP Retrials.....	71
5.3.1.2 Effect of Maximum Segment Size.....	74
5.3.1.3 Effect of Advertised Window Size (AWS)	75
5.3.1.4 Effect of File/Data Size.....	77
5.3.1.5 Effect of link capacity	80
5.3.2 <i>Performance of WWW</i>	85
5.3.2.1 Effect of Maximum number of RLP retrials.....	85
5.3.2.2 Effect of Maximum Segment Size.....	87
5.3.2.3 Effect of Advertised Window Size.....	89
5.3.2.4 Effect of Link Capacity	91
5.3.2.5 Effect of Data Size	92
CONCLUSIONS AND FUTURE WORK	96
6.1 CONCLUSION	96
6.2 SUMMARY OF RESULTS	97
6.3 FUTURE RESEARCH AREAS.....	98
REFERENCES	101

List of Figures

Figure 2.1	Wireline Network at Congestion.	14
Figure 2.2	Wireless Network Losses.	16
Figure 2.3	Wireless Lea.	17
Figure 3.1	File Transfer protocol (FTP).	36
Figure 3.2	TCP Connections for WWW. ..	37
Figure 3.3	SMTP Client/Server Applications.	38
Figure 4.1	Simplified Data Communication Model.	43
Figure 4.2	TCP Throughput if FER is applied to it directly.	55
Figure 4.3	TCP Throughput at different Parameter Settings.	58
Figure 4.4	Mean Delay Pattern.	60
Figure 4.5	Delay Comparison at different Parameter Settings.	61
Figure 4.6	Throughput Results form [8].	62
Figure 4.7	Delay Results from [8].	63
Figure 5.1	Effect of 2-, 3- RLP retrials.	72
Figure 5.2	Effect of RLP retrials for Burst of Errors.	73
Figure 5.3	Bar Representation of Maximum Segment Size Effect.	74
Figure 5.4	Effect of Advertised Window Size.	77
Figure 5.5	Effect of File Size on Throughput.	78
Figure 5.6	Combined Effect of AWS and File Size.	79
Figure 5.7	Bar Representation of AWS Effect with File Size.	79
Figure 5.8	Effect of Link Capacity.	83
Figure 5.9	Bar representation of Link Capacity against Throughput.	83
Figure 5.10	Combined Effect of AWS and Link Capacity.	84
Figure 5.11	Comparison of 2-, 3- RLP retrials.	86
Figure 5.12	Effect of 2-, 3-, 4- RLP retrials.	87
Figure 5.13	Effect of Maximum Segment Size.	87
Figure 5.14	Effect of Advertised Window Size.	90

Figure 5.15	Effect of Link Capacity.....	91
Figure 5.16	Effect of Variation in Avg. doc. Size.	93
Figure 5.17	Combined Effect of Doc. Size and AWS.	94

List of Tables

Table 3.1	Primary/Secondary Frames.	33
Table 4.1	Probability of TCP Packet in Error.	52
Table 4.2	Selected Simulation parameters.	57
Table 4.3	Comparison of TCP Throughput.	59
Table 5.1	Selected Simulation parameters and their Values.	69
Table 5.2	Effect of Maximum Segment Size on Throughput.	74
Table 5.3	Effect of increasing Receiver's Buffer.	76
Table 5.4	Effect of File Size on Throughput.	78
Table 5.5	Relationship between Link Capacity and RLP retrials.	80
Table 5.6	Effect of Link Capacity on Throughput.	81
Table 5.7	Effect of RLP retrials.	85
Table 5.8	Maximum Segment Size variations.	88
Table 5.9	Effect of Advertised Window Size.	91
Table 5.10	WWW Throughput Variations for Different Link Capacity.	92
Table 5.11	WWW Performance against Data Size. ...	93
Table 5.12	Doc. Size and AWS Effect on Throughput.	95

List of Acronyms

ACK	Acknowledgement
ARQ	Automatic Request
AWS	Advertised Window Size
BER	Bit Error Rate
BS	Base Station
CDMA	Code Division Multiple Access
CWND	Congestion Window
EIA	European Industries Association
ETSI	European Telecommunications Standard Institute
FEC	Forward Error Control
FER	Frame Error Rate
FTP	File Transfer Protocol
HTML	Hyper Text Multiprotocol Language
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IPCP	Internet Protocol Control Protocol
IS	Interim System
I-TCP	Indirect Transport Control Protocol
LAN	Local Area Network
LCP	Link Control Protocol
MH	Mobile Host
MODSIM	Modular Simulation Language
MT	Mobile Terminal
MSS	Maximum Segment Size
NAK	Negative Acknowledgement
POP3	Post Office Protocol
PPP	Point To Point Protocol
QoS	Quality of Service
RFC	Request For Comments

RLP	Radio Link Protocol
RTCP	Real Time Control Protocol
RTSP	Real Time System Protocol
RTT	Round Trip Time
SACK	Selective Acknowledgement
SMTP	Simple Mail Transfer Protocol
SNDCP	Sub Network Dependent Convergence Protocol
TCP	Transport Control Protocol
TDMA	Time Division Multiplex Access
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication System
WWW	World Wide Web

Chapter 1

Introduction

1.1 MOTIVATION

Data transmission services over mobile networks are one of the telecommunication sectors that are due to experience an important expansion in the near future. The potential demand on data mobile services is growing so much that it is forcing the telecommunication operators to supply this kind of communication. The reduction in size and prices of portable computers has extended its use in such a way that future networks are not conceived without integrating this mobile equipment to fixed computers. The use of applications like file transfer and e-mail, and in general access services to Internet, is driving portable users to demand them through the mobile link. In addition, the popularity of the Internet, coupled with the need to be connected to it anytime, anywhere, has seen a need to adapt these applications that Internet has to offer, to wireless computing. Internet, the biggest computer network, with millions of terminals, is based on the TCP/IP protocol architecture. The most used applications are e-mail, ftp, telnet and access to Web's, all of them are based on the TCP as transport protocol.

TCP/IP protocols were designed for being used in fixed networks and it was never thought that they would be applied in a new environment, which has very different characteristics. That is why a very determining aspect when evaluating data services based on TCP/IP over mobile networks, is the analysis of interaction between this protocol architecture and this new communication link.

There are specifically two problems that need to be addressed while Internet applications are to be transmitted in wireless environment:

1. The problem of roaming users and how they will be supported by TCP.
2. The problem that arises due to the fact that the wireless channels are not as reliable as the wireline channels.

These problems stem from the fact that reliable transport protocols such as TCP have been adapted to work well in traditional wireline networks where the packet losses occur mostly due to congestion in the network. When this occurs, TCP invokes the congestion control algorithms, which are intended to re-establish the network and avoid congestive collapse, and they do so very effectively. The implicit assumption in these networks is that the links are loss-less and hence do not cause significant amount of packet corruption. However, low throughput, high latencies, sporadic high bit error rates, and occasional temporary disconnection characterize communication in wireless networks. Mobility complicates matters further by increasing the complexity of routing data to machines that are not only away from their home network but also potentially changing their point of attachment as they move within cellular networks. This can cause packet losses, long delays as well as significant variance in packet delays. None of these losses require the invocation of congestion control, but that is how TCP reacts to them. It responds to all losses by invoking congestion control and avoidance algorithms by shrinking its transmission window and dropping data transmission rate even if there is no congestion in the network. This results in degraded end-to-end performance and wastage of network resources. Ideally, if a packet is lost due to handoff or corruption, the sending

TCP entity should retransmit immediately, rather than throttle back the data flow.

There are several studies on the aspects related to TCP performance over wireless links [2] [7] [18] [19] [20] [21] [22] [23] [24] [30]. In recent years, code division multiple access (CDMA) technology is capturing a significant interest in the realm of wireless communication services. The attractive features of CDMA are its potential capacity increase over other multiple-access methods, anti-multipath capabilities and its promise with respect to integration of data services (along with voice) in a bandwidth-on-demand, fair sharing manner. CDMA forms the basis of the IS-95 digital cellular standard, which was originally designed for voice application. Recently, TCP has been recommended as a major protocol suite for asynchronous data and fax communication over CDMA wireless links by the newly approved Data Services Option Standard of Wide band Spread Spectrum Digital Cellular System [11]. While the TDMA protocol guarantees reliable, in-order delivery of Link-Layer frames, the CDMA protocol only makes a limited attempt and leaves eventual error recovery to the (reliable) transport protocol. If TCP is applied directly to the wireless communication environment where channels are characterized by higher error rates, it may suffer severe performance degradation. Hence performance of TCP over CDMA, in particular, is a big issue. For the above reason, the standard introduces an additional link layer protocol, Radio Link Protocol (RLP), which stays below TCP/IP and above CDMA physical layer. In the context of data delivery, when the loss of link layer frames becomes significant, RLP plays a lead role in reducing the frame error rate (FER) to a level comparable with wireline networks. It operates on its automatic repeat request (ARQ) scheme to recover losses due to wireless channel errors and hides them from the transport layer. More

significantly, it is a negative acknowledgement or NAK-based finite selective repeat protocol. In this respect, RLP is a TCP-aware link layer protocol whose major purpose is to reduce frame error rate (FER) such that it will not trigger significant performance degradation at the TCP layer. In addition, TCP will invoke congestion control measures to eliminate all detectable errors for performance improvement.

Gang Bao [8] performed an analytical study to develop a relationship between frame error rate FER introduced by the physical layer of the IS 95 physical layer and the probability of a TCP segment in error after RLP has performed its maximum retransmissions to recover from the frame losses. This analytical study of [8] is the basis of our thesis to introduce packet errors at the TCP and evaluate the performance of Internet applications over the IS-99 protocol stack

1.2 THESIS CONTRIBUTION

Recommendations by the Data Services Option Standard of Wideband Spread Spectrum Digital Cellular Systems and introduction of the Radio Link Protocol provide a unique solution to the problem of TCP performance in a wireless environment. The effectiveness of RLP in recovering wireless generated packet losses has made it a part of upcoming wireless standards like IS-657 [27] and IS-707 [28]. While CDMA has been recommended as a choice of radio interface for third generation wireless systems, the performance of Internet applications using RLP needs to be evaluated which is the main focus of this thesis. Our contributions can be categorized in the following areas

- Evaluated proposals for improving TCP performance over Wireless
- Developed TCP/RLP test bed with IS-99 environment

- Built Traffic models for WWW and FTP/SMTP traffic patterns
- Evaluated the Performance of WWW and FTP/SMTP with
 - Variable Sized Segments
 - Effect of Maximum Number of RLP retrials
 - Effect of Maximum Segment Size
 - Effect of Advertised Window
 - Effect of File/Data size
 - Effect of Link Capacity

Two conference papers [39] [40] based on this study have already been accepted

1.3 THESIS ORGANIZATION

The rest of the thesis is organized as follows:

Chapter 2 provides an overview of the Transport Control Protocol and discusses the various algorithms that TCP invokes in case of network congestion and packet losses in fixed networks. A comparison of mobile environment is also included to get a better understanding of the problem. These discussions will also help in understanding the various behaviors of TCP, which need to be modeled.

Chapter 3 discusses the background where proposed solutions to make TCP behave better in a wireless environment are first discussed. This chapter also presents an overview of the Radio Link protocol and the traffic models for Internet applications.

Chapter 4 presents the design of test bed with a discussion of the behaviors of

TCP/RLP that were considered for the design. The test bed validation is also included.

Chapter 5 provides the simulation results and the experiments that were carried out to evaluate the performance of Internet applications.

Finally, Chapter 6 ends this thesis with the conclusion of the research and possible future areas of study are also identified.

Chapter 2

Overview of TCP

2.1 TCP OVER WIRELINE NETWORKS

The Transport Control Protocol (TCP) is a connection oriented inter-process communication protocol that provides end-to-end error, flow and congestion control protocol in the transport layer of the Internet suite. TCP was designed to work independently of lower layer implementations for data transport so it makes very few assumptions about the underlying network interface and uses unreliable datagrams service to send packets from one host to another. It introduces reliability by a conventional acknowledgment with retransmission scheme. The acknowledgment scheme is called "cumulative" because it reports how much of the stream has been accumulated [9]. TCP views the data stream as a sequence of octets or bytes that it divides into segments for transmission. A "segment" is the unit of transfer between TCP software on two machines. Segments are exchanged to establish connections, to transfer data, to send back acknowledgments, to advertise window sizes and to close connections. As a reliable transport protocol, TCP has to deal with two important problems of data communications:

- Efficient transmission
- Flow control.

2.1.1 Efficient Transmission

To achieve reliability, the TCP sender requires the recipient of a packet to

communicate with the source by sending back an acknowledgment (ACK) message as it receives data. TCP uses a specialized variable sliding window mechanism to provide an efficient transmission. As the variable sliding window protocol utilized by TCP keeps the network completely saturated with packets, it obtains substantially higher throughput than a simple positive acknowledgment protocol. Conceptually, the sliding window protocol always remembers which packet (in reality, it remembers the initial and final bytes number sent in a separate segment) has been acknowledged and keeps a separate timer for each unacknowledged data segment. TCP implementations maintain running averages of the delay between sending a data segment and receiving an acknowledgment of it, as well as an estimate of the expected deviation in this delay. This allows TCP to react to changing network conditions while trying to maintain maximum throughput, and the times are used to set the TCP retransmission timer. The timer is set to expire after the average delay plus twice the expected deviation. When the timer expires, TCP retransmits the segment on the assumption that the previous one was lost because of congestion. TCP also initiates congestion control mechanisms to allow the network to reestablish and avoid congestive collapse.

2.1.2 Flow Control

Having a mechanism for flow control is essential in an Internet environment, where machines of various speeds and sizes communicate through network and routers of various speeds and capacities. There are really two independent flow control problems to deal with.

The first problem justifies the requirement of a flow control mechanism that allows a

receiving machine to control the source that sends more traffic than it can tolerate or buffer. Flow control ensures that the sending terminal must not transfer data streams at a rate faster than the receiving terminal can absorb them. For this purpose, TCP allows the window size to vary over time and each acknowledgment from the receiver contains a window advertisement that specifies how many additional octets of data the receiver is prepared to accept. There should never be more unacknowledged data in flight than the receiver's advertised window. At any time, there is also a congestion window that TCP calculates dynamically.

The second problem deals with the end-to-end provision of flow control between the source and ultimate destination. When intermediate machines in the network become overloaded, the network is called in a "congestion" state and mechanisms to overcome the congestion conditions are called congestion control mechanisms. TCP implementations provide mechanisms for congestion control [34]. In fact, it makes use of its variable sized sliding window scheme to solve the end-to-end flow control problem.

In the following section, a brief discussion of TCP congestion control mechanisms is presented. These mechanisms are implemented in most versions of modern TCP including TCP New Reno and TCP Tahoe which are the most widely used versions of the Transport Control Protocol.

2.2 TCP RESPONSE TO CONGESTION

There are two indications of packet loss in a network.

1. Transmitter TCP time-out occurring
2. Receipt of duplicate acknowledgments from the receiver.

Modern implementations of TCP contain a number of algorithms aimed at controlling network congestion while maintaining good user throughput. These new algorithms are introduced to react against congestion and loss conditions.

V. Jacobson [13] proposed few algorithms for congestion avoidance and control. They include [12].

- Slow start
- Congestion avoidance
- Fast retransmit
- Fast recovery

2.2.1 Slow Start

The algorithm [12] operates by observing the rate at which new packets should be injected into the network is the rate at which the acknowledgments are returned by the other end. Slow start adds a new window to the sender TCP called "congestion window". When a new connection is established between two ends of TCP, the congestion window is initialized to one segment. The segment size is already negotiated between the two ends. Each time an acknowledgment is received, the congestion window is increased by one segment. The sender can transmit up to the minimum of the congestion window and the advertised window. The sender starts by transmitting one segment and waiting for its acknowledgment. When the acknowledgment is received, the congestion window is incremented from one to two and two segments can be sent. When each of these two segments are acknowledged, the congestion window is increased to four. This provides an

exponential growth in congestion window size, although it is not exactly exponential because the receiver may delay its acknowledgments by sending cumulative acknowledgment for more segments.

2.2.2 Congestion avoidance

The assumption of the algorithm is that the packet loss caused by the damage is very small, therefore the loss of a packet signals congestion somewhere in the network between the source and the destination. Congestion avoidance is an independent algorithm like slow start but when congestion occurs, TCP must slow down its transmission rate. In practice, congestion avoidance and slow start algorithms are implemented together.

While maintaining knowledge of the congestion window, TCP also maintains a slow start threshold size "ssthresh" for the congestion window. When congestion occurs, one half of the current window size (minimum of congestion window and receiver's advertised window) is saved in ssthresh. While performing slow start, when the congestion window reaches the size of ssthresh, congestion avoidance takes over from slow start. Congestion avoidance dictates that congestion window be incremented by $\text{segsize} * \text{segsize} / \text{congestion window}$ each time an acknowledgment is received instead of increasing the window exponentially as carried out in slow start. This is a linear growth of the congestion window, compared to exponential growth in slow start. The increase in congestion window should be at most one segment each round-trip time, regardless of how many acknowledgments are received in that round trip time (RTT) while slow start increments the congestion window by the number of acknowledgments received in a

round-trip time.

2.2.3 Fast Retransmit

TCP sends the data segments in order by indicating the sequence number of the data segments sent. The receiver TCP also receives the data in order by checking the received segment sequence number. It may happen that an unordered segment is received due to packet loss or some reordering of segments. The receiver TCP may generate an immediate acknowledgement (duplicate ACK) when an out-of-order segment is received. The purpose of this duplicate ACK is to let the other end know that a segment is received out-of-order and to tell what sequence number is expected. This duplicate ACK should not be delayed.

Since TCP does not know whether a duplicate acknowledgment is generated by a lost segment or just a reordering of segments, it waits for a small number of duplicate ACKs to be received. It is assumed that if there is just a reordering of the segments, there will be only one or two duplicates ACKs before the reordered segment is processed which will then generate a new ACK. If three or more duplicate ACKs are received in a row, it is a strong indication that a segment has been lost. TCP then performs the retransmission of what appears to be the missing segment without waiting for the retransmission timer to expire. This leads to higher channel utilization and connection throughput. Fast retransmit first appeared in the 4.3BSD Tahoe release of TCP.

2.2.4 Fast Recovery

A TCP sender invokes fast Recovery after fast retransmit [12]. When fast retransmit

sends the segment that appeared to be missing, congestion avoidance is invoked. This is fast recovery algorithm. The reason for not performing a slow start in this case is that the receipt of the duplicate ACKs tells TCP more than just a packet has been lost. Since the receiver can only generate the duplicate ACK when another segment is received correctly notifying that some data has left the network and is in the receiver's buffer. Also, there is still data flowing between the two ends and the TCP does not want to reduce the flow abruptly by going into slow start. The fast recovery algorithm was first introduced in 4.3BSD Reno release of TCP.

The fast retransmit and fast recovery algorithms are usually implemented together as follows.

1. When the third duplicate ACK in a row is received, set `ssthresh` to one-half the current congestion window but not less than two segments. Retransmit the missing segment. Set congestion window to `ssthresh` plus three times the segment size. This inflates the congestion window by the number of segments that have left the network and which the other end has cached.
2. Each time another duplicate ACK arrives, increment congestion window by the segment size. This inflates the congestion window for the additional segment that has left the network. Transmit a packet, if allowed by the new value of congestion window.
3. When the next ACK arrives that acknowledges the new data, set congestion window to `ssthresh` (the value set in step 1). This ACK should be the acknowledgment from the retransmission from step 1, one round-trip time after the retransmission.

Additionally, this ACK should acknowledge all intermediate segments sent between the lost packet and the receipt of the first duplicate ACK. This step is congestion avoidance, since TCP is down to one-half the rate it was at when the packet was lost.

2.3 WHAT IS DIFFERENT WITH A WIRELESS SYSTEM?

As discussed, TCP was designed to work with the wireline networks where the losses are due to congestion and the channel error rates are very low. Hence TCP assumes a relatively error free media and assumes that all packets losses are due to congestion so it reacts by invoking congestion control mechanism. To understand what causes degraded TCP performance over wireless environments, let us make a comparison of wireline and wireless networks.

Figure 2.1: Wireline Network at Congestion

In the current IP networks (Figure 2.1), the connection from the user to the correspondent server may go through a LAN, from the correspondent host, to a WAN (or over the Internet). Then the WAN is connected to a LAN and through the LAN to the user host. When IP packets travel from the correspondent host to the user host, the LANs will

have relatively much higher capacity to provide more bandwidth and usually need to support fewer users. In comparison, the WAN will have to accommodate much more traffic, sharing its capacity amongst many more users and allocating a slice of its bandwidth to each user. Traffic flows through several stages with different sized transport “pipes” over the WAN being the most narrow.

The bottleneck for traffic from the correspondent host to the user host (and from user host to the correspondent host) is located at both sides of the WAN (the situation with the Internet is similar). It is at these two points that most of the congestion happens for transporting IP packets. TCP/IP protocols have been optimized over many years for such wireline based networks. Packet loss is normally caused by congestion within the network, in particular over the WAN (or the Internet). However, packet loss is normally very low, typically around 10^{-9} BER (Bit Error Rate).

Wireless transmission has several distinctive characteristics from wireline transmission. It is unreliable, characterized by high error rate. It has limited and variable capacity relating to the availability of radio resources at any given time and location. It has many more variables that can be traded off against each other compared to wireline transmission. The radio transmission medium is unreliable (i.e., very high error rate) because of signal loss caused by path loss, fading, shadowing, and etc.

The network “hotspot”(refer Figure 2.2) is the two sides of the wireless link over the radio air interface compared to the other section of the network. The potential problems are much more pronounced. The wireless link has very restricted capacity in terms of the bandwidth and the number of users that it can support within a sector.

Figure 2.2: Wireless Network Losses

The characteristics of the wireless transmission can be summarized in Figure 2.3. Often, from the wireline network such as the Internet (or a WAN) to the wireless link, there is a reduction in the capacity of the transmission “pipe”, ie. the bandwidth and the number of users that may be supported. The pipe over the wireless link is also “leaky” in the sense that IP packets that travel through it may be lost over this link and that they will not arrive at the receiving end of the link (assuming we can ignore any undetected errors for now). The rate of this leak is much higher than in a wireline network based on high quality optical fiber or copper wire.

Over the time and space, the capacity may also shrink or expand. When a mobile host moves from one location within a wireless access network to another location (such as near the edge of the access network), it may be necessary to reduce the transmission rate because the radio signals may become weaker. When a mobile host is handed over from

one access network to the next, there is no guarantee that the same total amount of capacity will be available and what is allocated to a particular user or application may

Figure 2.3: Wireless Leak

also shrink. The access network changes (i.e. more high priority traffic such as voice comes in), it may be necessary to reduce the transmission rate of some or all data applications.

Data frames, by our transmission “pipe” analogy will literally leak from the pipe representing the wireless link before it reaches the destination. Part of the leak can be attributed to data frames that are in error on reception by the destination because of path loss, fading and etc. Part of the leak can be attributed to the data frames that never arrive because of shadowing (i.e., blockage of the mobile host by large objects). All this will add to the frame error rate (FER) which is the ratio of frames that do not arrive at the destination or received in error.

The Transport Control Protocol (TCP) in Internet Protocol suite responds to all losses (channel error losses, mobility generated losses or losses due to congestion) based on its

default implementation. For every packet loss, it assumes congestion in the network and invokes the congestion control algorithms. In this way, even if the network is not congested and losses are due to wireless characteristics, TCP slows down the transmission rate. This results in poor network resource utilization. TCP also performs retransmissions for each packet loss (conceptually, it is a byte loss but TCP sender has to retransmit the complete segment/packet containing the lost byte) that causes larger transmission delays and delay jitters.

Providing Quality of Service (QoS) for Internet traffic over wireless is a hot issue. Much research has been done in an attempt to improve TCP in mobile/wireless environments. Several solutions have been proposed, the most popular of these proposals are discussed with their practical applicability and usefulness in next chapter as background work.

Chapter 3

Background Work

The organization of this chapter explains the background work performed that finally led to the problem description and possible solutions. The background work is split in four sections. The first section discusses and evaluates the various approaches to improve TCP over wireless channels. Advantages and applicability of each approach are discussed. One of the proposals is selected for performance evaluation of Internet traffic. The second section provides an overview of the underlying Radio Link Protocol (RLP) proposed for over all TCP performance improvement. The next section provides the characteristics of Internet applications, which will be used in this study for their performance evaluation. Traffic models to generate Internet Traffic patterns will be highlighted in section 4. These traffic patterns are used with TCP/RLP test bed for final performance evaluation.

3.1 IMPROVING TCP OVER WIRELESS: PROPOSALS

There are mainly two different approaches for improving TCP performance in lossy environments. The first approach hides any non-congestion-related losses from the TCP sender and therefore requires no changes to existing TCP sender implementations. The intuition behind this approach is that since the problem is local, it should be solved locally, and that the transport layer need not be aware of the characteristics of the individual links. The second class of techniques attempts to make the sender aware of the existence of wireless hops and to make it realize that some packet losses are not due to congestion. The sender then avoids invoking congestion control algorithms when non-

congestion related losses occur. Finally, it is possible for a wireless-aware transport protocol to coexist with link-layer schemes to achieve good performance.

[1] Classified many proposed schemes into three basic groups based on their fundamental philosophy

- split-connection proposals
- link-layer proposals
- End-to-end proposals.

3.1.1 Split Connection Approach

Split connection proposals completely hide the wireless link from the sender by terminating the TCP connection at the base station. Such schemes use a separate reliable connection between the base station and the destination host. The second connection can use techniques such as negative or selective acknowledgments, rather than just standard TCP, to perform all over the wireless links.

3.1.1.1 Indirect TCP (I-TCP)

This scheme [2] involves splitting the TCP connection into two separate connections at the base station, one between the fixed host and the base station, and the other between the base station and the mobile host. Since the second connection is over a one-hop wireless link, there is no need to use TCP on this link. Rather, a more optimized wireless link-specific protocol tuned for better performance can be used. The advantage of this approach is that it achieves a separation of flow and congestion control of the wireless link from that of the fixed network.

This scheme suffers from three major drawbacks. First of all, TCP end-to-end semantics are broken. The base station fakes the mobile's TCP connections, and thus all of the acknowledgements that are seen by the fixed host are actually generated at the base station.

The second flaw is also related to the "faking" of the TCP connection at the base station. When setting up connections, it is necessary for the network applications on the mobile to create proxy connections at the base station. This is done using a new socket library that takes care of establishing communications with the base station and creating the TCP end-point there for connection to the fixed host. This requires relinking and possibly re-writing any TCP-based applications that would want to take advantage of the improved performance available using this method.

The third flaw is the software overhead created from the implementation of this approach. Every packet needs to go through the TCP protocol stack and incur the associated overhead four times (as compared to zero times for a non split connection approach), once at the sender, twice at the base station, and once at the receiver. This also involves copying of the data at the base station to move the packet from the incoming TCP connection at the outgoing one. Finally, since this approach maintains a significant amount of state at the base station per TCP connection, handoff procedures tend to be more complicated and slow.

3.1.2 Link-Layer Approach

Link-layer approach attempts to hide link related losses from the TCP sender by using local retransmissions and perhaps forward error corrections over the wireless links. The

local retransmissions use techniques that are tuned to the characteristics of the wireless link to provide a significant increase in performance.

3.1.2.1 The snoop Protocol

The snoop [3] protocol is a TCP-aware link protocol used to improve both handoff latency and TCP throughput. Handoff latency reduction is achieved by a routing protocol based on IP Multicast rather than Mobile IP. This protocol introduces a module, called the snoop agent, at the base station. The agent monitors every packet that passes through the connection in either direction and maintains a cache of TCP segments sent across the link that have yet not been acknowledged by the receiver. A packet loss is detected by the arrival of a small number of duplicate acknowledgments from the receiver or by a local timeout. The snoop agent retransmits the lost packet if it has it cached and suppresses the duplicate acknowledgements. In some sense it maintains the advantage of the split connection of the shielding the sender from vagaries of the wireless links and mobility of the receiver without sacrificing end to end semantics, modifying host TCP code, or re-linking existing applications. The base station routing code is modified by adding a layer, the snoop layer, which monitors every packet that passes through the connection in either direction. No transport layer code runs at the base station. Thus in effect, the base station hides the packet loss from the fixed host by not propagating duplicate acknowledgements, thereby preventing unnecessary congestion control mechanism invocation.

The beauty of using IP Multicast to reduce latency is that it avoids having to transfer the cache from one base station to next during handoff since this can significantly increase latency. It works by assigning the mobile a multicast address, and forwarding all

packets destined to it to that address (using encapsulation). In this respect, the protocol is similar to Mobile IP. However, rather than having the mobile change its care-of-address when it changes its point of attachment, it always keeps the same multicast address. Instead, what happens is that when the mobile is likely to handoff to a new base station, it requests that all potential recipients join the multicast group to which the mobile belongs. The group will likely have six to ten members and each of these will begin caching packets destined to the mobile. This ensures that every packet destined for the mobile will arrive at all base stations, allowing each to accumulate the cache that exists at the base station currently supporting the mobile.

This mechanism, though elegant, suffers from two major flaws. The first is that it is very insecure, anyone can join a multicast group, thus anyone who joins could receive packets intended for the mobile. The second flaw is that IP Multicast is still quite young, and as such, is not widely supported on the Internet. Thus, relying on it to provide general support for mobility is somewhat premature.

The major drawback of the snoop is its behavior in the presence of frequent handoffs. When a mobile host moves into a new cell, the new Mobile Service Station starts up a copy of the snoop module on behalf of the Mobile Host. This snoop module starts out with an empty cache and slowly builds this cache up. Thus, while the snoop module is building up its cache, the MH will see poor throughput. Since snoop cannot yet perform any useful retransmission or filtering functions. If the cell sizes are small (as they will be Pico-cellular environments), it is seen that the performance degradation can be serious. Apart from the above problem, snoop does not perform well in presence of lengthy

disconnections or in environments where there are frequent disconnections.

3.1.2.2 Radio Link Protocol for CDMA

Link-Layer protocols employ two main classes of techniques: error correction, using techniques such as forward error correction (FEC) and retransmission of lost packets in response to automatic repeat request messages. The digital cellular systems use either CDMA (Code Division Multiple Access) or TDMA (Time Division Multiple Access) as link layer protocols which primarily make use of ARQ mechanisms. While the TDMA protocol guarantees reliable, in-order delivery of Link-Layer frames, the CDMA protocol only makes a limited attempt and leaves eventual error recovery to the (reliable) transport protocol. Hence performance of TCP over CDMA, in particular, is a big issue.

For this reason, a Radio Link Protocol (RLP) has been introduced in the newly approved Data services option Standards for Wideband Spread Spectrum Digital Cellular System in a CDMA environment [11]. The Radio Link Protocol is a TCP-aware link-layer protocol where the major purpose of RLP is to reduce the Frame Error Rate (FER) to a level so that it will not trigger significant performance degradations at the TCP layer, which will eventually eliminate all detectable errors. In general, it is desirable to limit FER to the same level of a wireline network link.

RLP is a pure NAK-based finite selective repeat protocol. Since the number of allowable repeated trials is finite, it cannot completely eliminate detectable errors. When RLP at the receiving end finds a frame in error (or missing), it sends back a NAK requesting for retransmission of this frame. A timer is then set for the missing frame. When the timer expires for the first time, RLP resets the timer and sends back a NAK

twice. Each NAK received at the transmission end triggers a retransmission of the requested frame. When the timer expires for the second time, RLP resets the timer and sends back three NAK. This process continues until the number of timer expirations reaches a limit n ($n=3$ by default). RLP aborts the attempt after n retrials and passes whatever it has got to its upper layer. The final recovery of the missing frame is then achieved by TCP through its ARQ mechanism.

This approach has already appeared as a standard and plenty of research work is going on to integrate TCP and RLP. Recent standards of CDMA (IS99, IS657 and IS-707) are based on RLP specifications. However, the implementation of RLP needs many assumptions to be made when RLP interacts with TCP to provide an over all improved TCP/RLP throughput. A very careful understanding of how the two ARQs at TCP and RLP interact with each other is required.

3.1.3 End-to-End approach

End-to-End proposals attempt to make the TCP sender handle losses in the wireless links by distinguishing between congestion and other forms of losses and provide end-to-end QoS. There are many schemes suggested to deal with packet losses due to bit error rates and increased handoffs in wireless environments. We will present a discussion of some of the schemes here.

3.1.3.1 Fast Retransmit Approach

This approach [1] [4] [29] addresses the issue of TCP performance when communication resumes after a handoff. Unmodified TCP at the sender assumes that the

delay is caused due to congestion and when a timeout occurs, reduces its window size according to congestion control algorithms and retransmits its packets. Often, handoffs complete relatively quickly while long delays are required before time-outs occur at the sender and packets start getting retransmitted. An alternative end-to-end solution to the problem is for the transport protocol to resume communication immediately after handoffs complete. Modern TCP implementations including 4.3BSD Tahoe and TCP New Reno already perform similar fast retransmission when a sender receives three duplicate acknowledgements from the receiver. When activated, the fast retransmission procedure immediately retransmits the earliest unacknowledged packet, drops the transmission window and initiates the slow start algorithm. The rationale behind current fast retransmission is that three acknowledgments clearly indicate that packet loss has occurred and there is no need to wait for time-outs to occur before retransmission.

The disadvantage of this method is that it does not deal with the problem due to losses caused by the error characteristics of the wireless links. If used to deal with errors too, this would increase the amount of traffic and congestion in the fixed network, since every lost packet would have to be retransmitted from the source. Furthermore, there is no need to start the slow start recovery since there is no congestion on the new network and the transmission can go in the same speed as before and we would use fast retransmit algorithm.

3.1.3.2 Selective Acknowledgements

Since standard TCP uses a cumulative acknowledgment scheme, it often does not provide the sender with sufficient information to recover quickly from multiple packet

losses within a single transmission window. Several studies have shown that TCP enhanced with selective acknowledged performs better than standard TCP in such situations. SACKs [1] were added as an option to TCP by RFC 1072 [35]. However, disagreements over the use of SACKs prevented the specification from being adopted, and the SACK option was removed from later RFCs. Recently, there has been renewed interest in adding SACKs to TCP. Two relevant proposals are the recent RFC 2018 on TCP SACKs and the SMART scheme.

The SACK RFC 2018 [36] proposes that each acknowledgment contain information about up to three non-contiguous blocks of data that have been received successfully by the receiver. Each block of data is described by its starting and ending sequence number. Due to limited number of blocks, it is best to inform the sender about the most recent blocks received.

An alternate proposal, SMART, uses acknowledgements that contain the cumulative acknowledgment and sequence number of the packet that caused the receiver to generate the acknowledgment. The sender uses this information to create a bit mask of packets that have been delivered successfully to the receiver. When the sender detects a gap in the bit mask, it immediately assumes that the missing packets have been lost without considering the possibility that they simply may have been reordered. Thus, this scheme trades off some resilience to reordering and lost acknowledgements in exchange for a reduction in overhead to generate and transmit acknowledgements.

The main difference between SACK TCP and New Reno TCP is in the behavior when multiple packets are dropped from a window of data. Adding SACK to Reno and Tahoe

TCP does not change the basic underlying congestion control, avoidance and fast retransmission algorithms. The results show that the addition of SACK allows additional improvements to TCP, in addition to improving the congestion control behavior when multiple packets are dropped in one window of data.

3.1.3.3 Modification to TCP as suggested by the University of Maryland

This approach [5] does not have any specific name but has been suggested by researchers at the department of Computer Science at the University of Maryland. In this approach, it is suggested to modify TCP for improved end-to-end delay and latency characteristics in wireless networks, while maintaining end-to-end semantics. The basic wireless model that is considered is that of a source host which is connected to a base station by a fixed wireline network and a wireless host which is just a single hop from the base station.

There are a few important observations/assumptions that the researchers made while proposing this solution. First of all, any packets loss between the source and the base station is solely due to network congestion and any packets dropped by the wireless link are due to channel noise and Bit Error Rates (BER). The second assumption is that the connecting base station of the wireless host does not change, i.e the scenario of handoffs is not considered. Also, it is assumed that the source is aware that the other end is a wireless host.

There are few basic modifications made to TCP for the purpose of better ensuring that the source can distinguish between congestion in the fixed network and the noise in the wireless link as the cause for packet loss. The changes at the receiving side (the wireless

host) are to ensure that in case packets are dropped due to noise in wireless channels, the source starts to retransmit packets earlier than its timeout deadline for which it normally waits.

When a TCP connection is to be opened between a fixed host on a network and a wireless host, it opens two independent connections, one to the base station and the other to the mobile host. The call setup functionality is left unchanged as is in the original TCP. The connections between the source and the base station basically serve as a control to estimate the network congestion that occurs between two ends. With the assumption that the mobile host receives all packets through the base station, it is expected that packets on both of these connections could be routed by almost the same route and would see the same network congestion. Packets on the control connection are sent at the some regular intervals but at rates sufficiently low so as to cause too many overheads in the network because of these control packets. The TCP source will also be sending packets on the connections with the mobile host as the application hands down packets to it. By comparing the fraction of the packets that are acknowledged within the timeout on the two connections, the source will determine if there is congestion in the network, which is causing the packets to be lost, or it is the channel noise. If the acknowledged fraction is significantly different in the two cases then the conclusion is that the channel noise is causing the packet loss. In this situation, the source will be prevented from reducing the congestion window size and it will continue its linear increment at the same rate. If the two fractions are the same, then the normal congestion control mechanisms are allowed to proceed as it would in ordinary TCP. At the wireless host side, to pre-empt the source to re-send the packets that are lost due to channel noise (or even otherwise), the technique of

fast retransmission is employed. The source on seeing a duplicate acknowledgment decides that its previous packet sent have been lost or corrupted. Hence it retransmits the subsequent packet even before the timer expires and restarts the timer.

Although the above approach is indigenous, the drawbacks in the scheme can easily be identified. First of all, the source has to have the prior knowledge that the destination is wireless that is not practical. Secondly, apart from host nature, two connections for each TCP connection have to be opened which is an unnecessary overhead to the network.

3.1.3.4 Mobile TCP

Mobile TCP [6] aims at examining the several aspects of performance and connection management during handoffs and interface switching in mobile computing environments. The motivation behind this protocol is that it is insufficient for network layer mobility protocols to preserve high performance operation of reliable transport protocols such as TCP under certain circumstances. Hence, it is necessary to introduce the notion of mobility on the transport level (Mobile TCP) to maintain robust operation despite lousy mobile activity. The reasons for introducing mobility in the transport layer as attributed by the researchers are as follows.

Mechanisms that depend on Mobile IP for transparent mobility management will necessarily have to work within the constraints of triangular routing(i.e a packet that has to reach the mobile host may always have to pass through the home agent and then tunneled to the mobile hosts care of address as specified by Mobile IP. The congestion window of a TCP connection will therefore be affected if there is congestion in the region

of the home agent, even if there is no congestion in the path between the correspondent host and the mobile host. Furthermore, both the Round Trip Time (RTT) and the variation in the Round Trip Time of a connection will increase.

As the number of overlapping wireless network providers increase, it will be possible and even desirable to switch between network interfaces and service providers in order to overcome congestion, fades, and bursty channel errors (similar to on-the-fly channel hopping in cellular telephones). It has to be noted that different service providers will assign different home addresses for reasons of accountability and administration. The researchers argue that there is no reason why a transport connection cannot span multiple network connections with different end point addresses (the requirement of a single end-point network address for a transport connection is an artifact of the design of TCP/IP). Hence it is proposed to provide connection management in Mobile TCP in order to handle both the problems with triangular routing and the ability to switch interfaces with different home addresses.

Another important design feature of Mobile TCP is that when packets are actually lost due to handoff or interface switch, retransmission should be allowed to occur without congestion control backoff and the resulting performance degradation. Furthermore, handoffs between two neighboring cells of the same network (as in the case of a wireless LAN) may be able to use the same value of Congestion Window size (cwnd) and Round Trip Time (RTT) values.

The Mobile TCP approach is different from all other protocols discussed in the sense that it tries to address the problem of roaming users and assumes to a certain extent the

responsibility of the Mobile IP. While this may be a good idea, it deviates from the proposed standards and does involve modification to TCP at the remote station. Moreover, duplication of work may happen in IPv6 since it alleviates the problem of triangular routing.

3.1.4 Selection among the Proposals

The default implementations of the Transport Control Protocol (TCP) prevents it from performing effectively over wireless channels. A great deal of research work is being carried out to tune TCP with wireless characteristics. Most popular of the modern approaches have been described with detailed discussion of the advantages and practical application of each scheme. For any proposal to be globally applicable, it is necessary that the scheme should not violate the end-to-end TCP semantics. Also, no major modifications should be proposed in base Transport Control Protocol that is already existing. Most of the proposals discussed yet have failed to maintain TCP end-to-end semantics or they propose major modifications in TCP implementations. Schemes like Fast Retransmit and SACK options are widely accepted as modifications in TCP but these modifications are suggested for improving its performance over fixed networks for heavily congested condition or for differentiate loss of packets with their disordering. That is why, new versions of TCP over fixed networks include these modifications by default. In wireless channels, the problem is not misordering of received segments (data bytes) but these are the losses due to wireless characteristics responsible for creating performance degradations at the TCP level. For these reasons, SACK and Fast Retransmit are not enough to improve TCP performance in wireless networks.

The specifications of EIA/TIA/IS-99 do not suggest any modification in TCP implementations. In fact, the implementations of TCP are kept the same as defined in RFC 793 [25]. For performance improvement, it introduces an independent middle layer protocol called Radio Link Protocol (RLP) that operates on its own ARQ to recover the losses generated by the channel errors. The standard also declares complete specifications of the Radio Link Protocol (refer to the next section). Also, it uses CDMA as radio access interface for mobile users which has been defined as the radio interface for third generation wireless networks. A great deal of research work is in progress to define a network architecture using CDMA as a radio access technique. Considering all these factors, any study of Internet performance evaluation using the IS-99 standard protocol stack will be meaningful.

3.2 OVERVIEW OF THE RADIO LINK PROTOCOL

Radio Link Protocol layer is defined as a middle layer, which sits over IS-95 physical layer. It provides data stream services to its upper layer in both forward and reverse CDMA channels. In general, RLP breaks down an upper layer packet into a number of

Primary traffic		secondary Traffic	
Full rate	171	Rate 1	168
Half rate	80	Rate 7/8	152
1/8 rate	16	Rate 3/4	128
		Rate 1/2	88

Table 3.1: Primary/Secondary Frames

RLP frames when the packet is bigger than a RLP frame. RLP frames may also have variable lengths especially when they share time slots with another traffic stream. The number of bits contained in a frame is therefore a function of traffic type as shown in

Table 3.1. RLP works on NAK-based selective repeat ARQ mechanism. The receiver does not acknowledge correct RLP data frames. Instead, it only requests the retransmission of RLP data frames that were not received or that were lost.

When it finds a frame in error, it sends back a NAK requesting for retransmission of this frame. A timer is then set for the missing frame at the receiver end. When the timer expires for the first time, RLP resets the timer and sends back NAK twice. Each NAK received at the transmission end triggers a retransmission of the requested frame. When the timer expires for the second time, RLP resets it and sends back three NAKs. This process continues until the maximum number of timer expiration reaches its limit. At this stage, RLP aborts the attempt and passes whatever it has got to the upper layer. The final recovery of the lost frame is achieved by the reliable transport protocol by its ARQ mechanism.

RLP maintains a sending sequence counter $V(S)$ and two receiving sequence counters $V(R)$ and $V(N)$. $V(S)$ is incremented by modulo 256 whenever a new frame of non-zero bytes is sent out. $V(R)$ contains the sequence number of the next expected new frame. $V(N)$ is the oldest sequence number of the frames.

For a valid received data frame containing a non-zero number of data octets, the RLP shall compare its sequence number (M) to $V(R)$ and $V(N)$

- If $M < V(N)$, discard it (duplicate frame).
- If $M = V(N)$ update $V(N)$ to the next oldest missing frame sequence number.

Pass received frames up to $V(N) - 1$.

- If $V(N) < M < V(R)$, store frame M into the re-sequencing buffer if it is

missing.

- If $M=V(R)=V(N)$, pass all received frames up to $V(R)$ to the upper layer.
- If $M > V(R)$, store frame M into the re-sequencing buffer.
- For all cases, send NAKs of missing frames

3.3 INTERNET PROTOCOL APPLICATIONS

The present work on wireless IP performance assumes that the applications are still those prevalent in the wireline networks. These applications include Web browsing using the HTTP protocol, bulk data transfer using FTP, email transport and access using SMTP and POP, streaming audio and video using RTSP and other multimedia applications such as NetMeeting using RTP/RTCP and H323 protocols. Although it is expected that new applications will emerge, these protocols will support most of the applications in the future. HTTP, FTP, SMTP and POP3 are built on top of the transport layer protocol TCP. RTSP and RTP/RTCP are built on top of UDP (RTSP may also be run on top of RTP/RTCP). As the study is based on the performance evaluation of the applications based on TCP as a transport protocol, the focus will be on the performance of the applications like web Browsing, FTP and SMTP. In the following sections, the characteristics of Internet applications being considered for performance evaluation will be discussed briefly.

3.3.1 File Transfer Protocol (FTP)

The File Transfer Protocol [33] is the Internet standard for file transfer. It has been selected as one of the priority applications for performance evaluation.

FTP uses two TCP connections (control and data connections) to transfer files between a client and a server as shown in Figure 3.1. It has a connection setup phase, during which it connects a client to a server in the normal client-server fashion. During this phase, it resolves IP addresses, permits user login, and sets up the control connection. This connection stays up for the entire time that the client communicates with the server

Figure 3.1: File Transfer Protocol (FTP)

with commands such as “open”, “ls”, “cd”, and “get”.

After connection set-up, FTP waits for user commands while it maintains its control connection. Certain commands (e.g., “get”) cause FTP to create a data connection between the client and the server. This TCP connection performs the actual data transfer, which is established using TCP bulk transfer mode. This is the data transfer phase. There can be only one single data connection at one time. It terminates after the data is transferred. At this stage, TCP passes control back to the control connection. In the termination phase, TCP shuts down the control connection. And exits.

3.3.2 Web Browsing (WWW)

HTTP [37] is a network protocol used to deliver virtually all files and other data on the World Wide Web such as HTML file and image files. The Web client communicates with a Web server using one or more TCP connections (Figure 3.2).

HTTP will establish its first TCP connection and retrieve the HTML document

Figure 3.2: TCP connections for WWW

identified by a URL. After the document is retrieved this TCP connection is terminated. What happens next depends on the client browser. Typically, the client browser will extract the embedded HTML document entities from this first document and retrieve them based on extracted URL's.

3.3.3 Electronic Mail (SMTP)

Electronic mail is the most popular application used over the Internet. SMTP [38] is used to transmit email from the sender to the next to last hop of the recipient. The recipient will use POP3 to download their email at the last hop.

SMTP (Simple Mail Transfer Protocol) is independent of the particular transmission

Figure 3.3: SMTP Client/Server Application

subsystem and requires only a reliable ordered data stream channel (Figure 3.3). As a result of a user mail request, the sender SMTP establishes a two-way transmission channel to a receiver-SMTP. The receiver-SMTP may be either the ultimate destination or an intermediate

3.4 TRAFFIC GENERATORS

In order to analyze the performance of Internet applications, traffic generators that can provide data to our TCP/RLP simulation test bed are required. These generators should generate the traffic of required characteristics representing the pattern of data arrival from a particular application. Traffic generators that are based on selected models from different sources are modeled. For FTP and SMTP, the behavior of data arrival is taken from traffic models discussed in the ETSI standard for UMTS [17]. The document presented a traffic model for circuit switched data services. WWW traffic pattern is based on the model discussed in [14]. In the following section, we will discuss briefly our traffic models.

3.4.1 WWW Traffic Generator

The WWW model is selected from one the theories [14] [15] [16] because the model

is developed from actual data collected over 20,000 document requests by 793 active users during a two day period.

Suang Deng [14] describes the data traffic characteristics of individual subscribers on the future network based on today's WWW traffic. When a user requests a page, several URL requests may follow. The first one transmits the user's direct request to the server. The other requests may be automatically generated by the browser program, e.g., each in line image (e.g., logos, icons, buttons, etc) requires a separate request be sent automatically by the client program during the download of a page.

An ON-OFF process rather than the simple Poisson process may model this behavior. The active, or ON period, consists of series of requests which follows the inactive OFF period. The wide data collection in a real environment (GTE Labs) supports the validity of the above model.

The complete description of this model is as follows.

3.4.1.1 Traffic Model

Based on the work in [14], the source model can be described by the distributions of three random variables:

1. x : the duration of an ON period in seconds
2. y : the duration of an OFF period in seconds
3. z : the inter-arrival time of requests during an ON period in seconds.

The following section describes briefly the results in [14], which are used to build the WWW traffic model.

3.4.1.1.1 ON PERIOD DISTRIBUTION

- The exact boundaries of ON and OFF periods are dependent on the definition of the minimum OFF period.
- A sequence of document requests with inter-arrival time less than 60 seconds is considered to form an ON period, and the occurrence of a request more than 60 seconds after the previous request indicates an OFF period. Two consecutive OFF periods separated by a single request are combined into one. It was observed that the distribution is insensitive to the particular choice of the OFF period threshold.
- The actual data collected appears to belong to the Weibull distribution, whose probability density function is given by:

$$p(x) = k/\theta (x/\theta)^{k-1} e^{-(x/\theta)^k}$$

- The probability distribution function of Weibull distribution is as follows:

$$F(x) = P(t \leq x) = 1 - e^{-(x/\theta)^k}$$

- The values of k and θ in [14] are suggested as:

$$\theta = \exp(4.5)$$

$$k = 0.88$$

Where x is in seconds.

3.4.1.1.2 OFF PERIOD DISTRIBUTION

- It represents the thinking time of the user, which results in a significant pause in the communication activities.

- Requests with inter-arrival time interval longer than 6000 seconds can be considered as two separate sessions.
- The empirical results suggest that the duration of the OFF period belong to Pareto distribution having the following probability density function:

$$p(x) = \alpha k^\alpha / y^{\alpha+1}$$

- The probability distribution function of a Pareto distribution is as follows:

$$F(y) = P(t \leq y) = 1 - (k / y)^\alpha$$

- The values of k and α for WWW types of traffic in [14] are as follows

$$k = 60$$

$$\alpha = 0.5$$

Where 'y' is in seconds.

3.4.1..1.3 INTER-ARRIVAL DISTRIBUTION

- The empirical results show that the inter-arrival distribution, $F(z)$, is given by a Weibull distribution again. However, the values of k and θ are different from those that were used in the calculation of the ON period. These values for OFF period are suggested in [14] as:

$$k = 0.5$$

$$\theta = \exp(1.5)$$

3.4.2 FTP/SMTP Generator

ETSI recommended traffic models for third generation European systems UMTS for wireless technology. Traffic models for FTP/SMTP are taken from these recommendations [17]. The standard states that circuit switched data services like fax, e-mail, file transfer can be modeled by a constant bit rate service with 100% of activity during call duration period. The call duration period depends on the size of data file to be transferred and the link capacity.

Chapter 4

Design and Validation of TCP/RLP Test Bed

4.1 Simulation Environment

The Simulation test bed is based on a simplified model of wireless CDMA data communications protocol stack proposed in [11]. In our setup, we have a mobile terminal exchanging data with a base station on a wireless link employing CDMA digital communication technique. The data flow is considered from Base Station (BS) to Mobile Terminal (MT). The specifications of the wireless channels are selected from the options

Figure 4.1: Simplified Data Communication Model based on [11]

provided in IS-99 [11]. The wireless link capacity is assumed to be 9.6 kb/s and CDMA channels carry only primary full rate traffic. The protocol stack on both sides (MT and BS) is symmetric which is in practice and is shown in Figure 4.1.

The purpose of the simulation setup is to analyze the effect of RLP presence on

overall performance of TCP. The performance attributes of interest in the setup include the overall TCP/RLP throughput and the mean one way propagation delay required to transmit data from Base Station to Mobile Terminal. We define throughput as the total data bytes that have been received correctly at the destination TCP in a unit of time. Mean delay is defined as the average time that data segments have to incur a while transmitted over the wireless link to be received by the destination. The two important components of the test bed are TCP and RLP ARQ mechanisms and the overall system performance will largely be dependent on the interaction of these two ARQ.

The physical layer of the CDMA environment introduces channel errors with specific Frame Error Rate (FER) depending on channel characteristics. If there is no RLP implemented in the protocol stack, the Transport Control Protocol will have to deal with all errors generated by the wireless environment. With its default implementation, TCP reacts to every packet error or packet loss as if there is congestion in the network. It slows down the transmission rate, makes retransmissions to prevent the network from falling into congestive collapse and provides a reliable transmission. This operation degrades the Quality of Service (QoS) severely, resulting in poor network resource utilization, larger delays and delay jitters. With RLP operating below TCP/IP, all channel errors will be transferred to RLP frames, RLP attempts to recover the errors through its own ARQ before everything is delivered to the upper transport layer for final recovery. As we will see, this results in reduced probability for TCP to receive packets in error and contribute to the overall TCP performance improving substantially.

Gang Bao [8] presented an analysis at RLP layer in which he developed a relationship

between the Frame Error Rate (FER) introduced by the physical wireless channels to RLP and the probability of a TCP packet in error for a given number of RLP attempts to recover erroneous frames. We try to utilize the results of this study to introduce Packet Errors at TCP layer against different settings of fixed Frame Error Rate. Our simulation test bed then models the behavior of TCP under normal and in error conditions. The overall performance of the TCP/RLP model is then analyzed by varying few parameter settings at TCP and RLP level. We validate our simulation test bed by comparing our results to the study carried out by [8]. As a contribution, we will extend our study by analyzing the behavior of real traffic patterns by applying those traffic patterns to the test bed keeping their individual QoS requirements in consideration.

In the following section, we discuss TCP/RLP test bed design and try to present different network conditions and how TCP reacts to those network variations with its default implementations. The calculations of performance parameters and their modeling in the simulator design are also discussed in the end. The performance results are plotted and compared with the results presented by [8] to justify the consistency of the design under varying network conditions.

4.2 MODELING TCP BEHAVIOR

- The simulation test bed is based on IS-99 standard [11] with few assumptions made to simplify simulation environment.
- First of all, ICMP, IP, SNDCCF, IPCP, LCP and PPP protocols are assumed to be acting as an interface (Fig 4.1) because these protocols generate a relatively fixed amount of overhead to overall system performance that can be included

easily when required.

- Procedures related to TCP connection setup and connection tear down are not included in the simulation model.
- When the system is in operation, two types of data traffic could occur. A single data stream in the forward link and the short control packets such as ACK packets in the reverse link. We assume that there is no data or voice traffic in the reverse link. Since ACK packet is much smaller than a TCP packet at the forward link, the reverse link is assumed to be error free for simplicity.
- The data traffic for the forward link is modeled by an infinite buffer queue carrying infinite number of packets so TCP always finds data packets of size MSS to be transmitted according to the allowed window size

The parameters required for the TCP model are based on RFC 793 [25] with major modifications suggested in IS-99 specification [11], which states:

- The maximum Segment Size (MSS) should not be smaller than 536 bytes (536 bytes are default).
- The transmitted segment size should not be bigger than 2047 bytes.
- TCP advertised window should not be smaller than 2 MSS and should not be bigger than 4 MSS.

4.2.1 Maximum Segment Size

The selection of the Maximum Segment Size (MSS) is limited to recommendations in

[11]. The default value of MSS is taken as 536 bytes. It is an important parameter in computing the overall system performance, therefore performance variations against MSS are computed. We performed simulations with different settings of Maximum Segment Size within the range of specifications.

4.2.2 Maximum Window Size

The size of the TCP congestion window is computed on the basis of the Advertised Window Size. According to RFC 2001 [12], TCP can advertise window size up to 65535 bytes but we are limited in a range of 2MSS to 4MSS as specified for the TCP model [11]. For simplicity, it is assumed that the advertised window size is always greater than the TCP congestion window.

4.2.3 Window Initialization

When a new session is opened, TCP starts data transmission with an initial window size of 1 MSS (as specified in RFC 2001 [12]), and then expands its window size following the ACK received and the advertised window size.

4.2.4 Congestion Window Management

It has already been stated that TCP operates on a sliding window mechanism and the size of TCP congestion window varies based on the network conditions. In fact, the sending entity keeps adjusting the window size as it receives acknowledgements from the receiving end. TCP congestion window management is a difficult behavior to model. In our simulation test bed, we introduced Flags to control the sliding window operation of TCP. These flags are set and reset with the varying network conditions and receiver

responses. The complete model of window management can be summarized with the following behavior-oriented categories of TCP operation, which it exhibits in response to acknowledged data.

4.2.4.1 Normal operation

The *CongestionFlag* is defined to determine the present network condition. While transmitting data, the TCP sender keeps a record of data in flow and data expected to be acknowledged next. When an acknowledgement arrives, the TCP sender compares it with the expected acknowledgement. If the received ACK is the same as it was expected, *CongestionFlag* is set to *FALSE*. This indicates a successful reception of previously sent data. In these scenarios, TCP expands its congestion window for each successful acknowledgement until it finds the threshold value *ssthresh* (for details see TCP overview in chap. 3). The threshold value is updated after every network congestion notification. For a new session, *ssthresh* is initialized to the maximum allowable window size.

4.2.4.2 Timer Time Out occurs without receiving ACK

For each segment transmitted, TCP sets a timer. The acknowledgment for this segment should arrive within the timer time out. On the other hand, ACK for a sent segment can be delayed due to network conditions (congestion or channel errors). If TCP finds its timer expired without an ACK received, it declares congestion in the network and invokes congestion control algorithms (for details of congestion control algorithm, refer to chapter 2 or [12] [13]). On congestion notification, TCP retransmits the lost data packet and tries to shrink its transmission window size to reduce the amount of data entering the network. In our simulation test bed, the *CongestionFlag* is set to *TRUE* in

response to any delayed ACK that notifies congestion. The congestion window controller shrinks the window size to 1 and updates threshold value *ssthresh*. TCP recalculates the threshold value *ssthresh* by reducing it to half the size of the congestion window size recorded when congestion occurred. It restarts transmission with a window size of 1. This operation represents the slow start algorithm. TCP also re-estimates the Round Trip Time (RTT) , makes an exponential back off by doubling its timer time-out value. After transmitting the segment, it waits for the ACK and then reacts accordingly. If everything goes normal, and the congestion window expands up to the new threshold value, congestion avoidance algorithm is invoked. During this phase, TCP expands its congestion window linearly.

4.2.4.3 Three Duplicate ACKs received

TCP may generate an immediate acknowledgement (a duplicate ACK) when an out-of-order packet is received. This duplicate ACK should not be delayed. The purpose of it is to let the other end know that a packet was received out of order, and to tell it what sequence number is expected. At this stage, TCP invokes fast retransmit and fast recovery algorithms.

When the third duplicate ACK in a row is received, TCP sets *ssthresh* to one-half the current congestion window but not less than two segments. It then retransmits the missing segment, sets the congestion window to *ssthresh* plus 3 times the segment size. This inflates the congestion window by the number of segments that have left the network and which the other end has cached. Each time another duplicate ACK arrives, TCP increments the congestion window by the segment size. This inflates the congestion

window for the additional segment that has left the network. The rest of the transmission is followed according to congestion window size. With the acknowledgement of new data, the congestion window is set to the threshold value *ssthresh*. This step is congestion avoidance. In this way, TCP performs congestion avoidance without going into slow start. The simulation test bed defines *fastRetransmitFlag*, which notifies the transmitting entity about three duplicate ACKs. TCP then invokes fast retransmit and fast recovery algorithms.

4.2.5 Communicating Transmitting and Receiving Ends

The whole operation of TCP is a collection of responses while the two ends of TCP are communicating with each other. The TCP sender has to manage the window size in response to receiver's generated acknowledgements while the receiver end is generating acknowledgements based on the data segments it receives. So a well-synchronized and correlated operation between the two interacting entities of TCP is necessary for a true TCP system.

This issue is addressed carefully in simulator design. Providing separate link lists at each of transmitting and receiving entities synchronizes the operation between the two ends of TCP. The data transmitted by the sender is recorded as Segment Objects in a link list named *SegmentQueue*. Each segment object in the list carries its sequence number, transmit time and segment size with it. The contents of *SegmentQueue* are accessible by the Receiver Object. In each cycle of data transfer, Receiver Object reads the contents of *SegmentQueue*, which provides the sequence numbers of currently generated segments that are unacknowledged yet. For each sequence number obtained from the list, receiver

verifies the correct sequencing and in-order delivery and then generates an ACK correspondingly. If finds any segment out of order or missing, it either generates a duplicate ACK or it does not generate an ACK for the lost or missing segment number. The acknowledgements generated by the receiver end are recorded in a separate link list, *ackQueue* as ack Objects. Each ack Object carries the acknowledgement number and the segment received time. The contents of *ackQueue* are readable by the transmitting end of TCP, which matches each ACK in *ackQueue* with the expected acknowledgement record it maintains. If it finds any mismatch or observes duplicate ACKs in the link list, necessary decisions are taken to immediately invoke appropriate congestion control algorithm. In this way, a perfect synchronization is established between the two ends of TCP.

4.2.6 Introducing Errors

As mentioned before, the Frame Error Rate (FER) of the wireless channel is introduced to the RLP layer, which attempts to recover the losses before it transfers data to the reliable transport protocol for final recovery. Gang Bao [8] has presented an analysis in which the probability of a TCP packet in error is shown as a function of the Frame Error Rate (FER) of the wireless channel and the number of attempts or retrials that RLP makes to recover those wireless generated losses. Assuming a fixed Frame Error Rate (FER) introduced at the RLP, the probability of TCP packets in error (after given RLP retrials) can be calculated as

$$P_e = 1 - (1 - p((2 - p)p)^{(n/2 + n/2)L})$$

Where, ' P_e ' is the probability of a TCP packet in error. The fixed Frame error Rate

(FER) is denoted by 'p' and the maximum number of attempts or retrials made by RLP by

number of RLP retrials	FER (%) 1	FER (%) 2	FER (%) 3	FER (%) 4	FER (%) 5
1	5.77E-03	2.29E-02	5.14E-02	9.01E-02	1.32E-01
2	2.28E-06	3.60E-05	1.79E-04	5.58E-04	1.34E-03
3	1.80E-11	2.23E-09	3.70E-08	2.69E-07	1.25E-06

number of RLP retrials	FER (%) 6	FER (%) 7	FER (%) 8	FER (%) 9	FER (%) 10
1	1.89E-01	2.41E-01	3.03E-01	3.63E-01	4.27E-01
2	2.74E-03	4.99E-03	8.37E-03	1.32E-02	1.97E-02
3	4.33E-06	1.23E-05	3.05E-05	6.73E-05	1.36E-02

Table 4.1: Probability of TCP Packet in Error

'n'. Each TCP packet is to be segmented into number of RLP frames. The factor 'L' denotes the number of RLP frames generated from one TCP packet. This number is dependent on the type of the traffic and the capacity of CDMA channels. With a channel capacity of 9.6 kb/s, an RLP operating at the full rate of primary traffic carries 19 bytes per frame (IS-99 specs. [11]). The implicit assumption of fixed Frame Error Rate (FER) at RLP is reasonable in general since power control mechanisms implemented at CDMA physical layer can often keep FER around 1% under normal conditions. Table 4.1 above presents the probability of a TCP packet in error against Fixed FER values introduced at RLP layer and after number of retrials by RLP to recover those errors.

We used these results to introduce errors in TCP packets. For this purpose, an error model called *RLPErrorModel* is developed. It defines a local variable *errorCount*, which keeps track of the number of segments and is incremented for each segment transmitted. Another variable, *errorThreshold*, contains the value of error rate. This value is a function of FER as shown in Table 4.1. During data transmission, the *errorThreshold* is compared with the *errorCount* for each segment generated. If the two parameters are found to have

the same value, a data segment in error is introduced. This segment also carries its segment number and transmission time. The *SegmentQueue* does not add any segment in error, thus creating a missing segment number or a lost packet within the link list. When the receiver object queries *SegmentQueue*, it finds a missing segment and acts accordingly. This action can be generation of duplicate ACKs or a delayed ACK for the missing or lost segment. When the transmitting end finds duplicate ACKs or a timer time out, it invokes its congestion control algorithms.

4.2.7 Statistics Collection

As mentioned before, the performance parameters of interest are the overall TCP throughput and the Mean delay experienced by the segments. We computed these performance parameters as a function of other parameters of interest at TCP and RLP layer. The Maximum Segment Size (MSS) at the TCP level and the maximum number of RLP retries at the RLP layer are considered to be the two most important parameters in determining overall system performance in [8]. We simulated different combinations of these parameters to observe their effect and the trend of system response performance.

In the following sections, we try to explain how we modeled our performance parameters and how the simulation test bed is used to collect statistics.

4.2.7.1 TCP Throughput

The overall TCP throughput is defined as the total number of data bytes acknowledged error free by the receiver in a given time.

$$\text{TCP Throughput} = \text{Data Bytes Received} / \text{Total Simulation Time}$$

We have already discussed that TCP has different behaviors for different network conditions. If the network is not congested or the number of errors at the TCP layer is small, more data will be transmitted and received correctly, and ultimately overall TCP throughput will be high. In contrast, for more errors, TCP will have to make more retransmissions, and most of network resources will just be utilized in re-transmitting the same packet. This will result in a degraded TCP performance.

4.2.7.2 Mean Delay

Mean delay is defined as the average time taken by a data segment before it is received correctly at the destination. It is an important performance parameter when real time or time-sensitive data are considered over the Internet Protocol. In a wireless environment, TCP introduces long delays and delay variations because of its increased retransmissions in response to increased number of channel errors. We describe here how the delay calculations are carried out in our simulation environment stating all our assumptions.

To simplify calculations, a few assumptions are made about different factors that contribute to the Mean delay calculations.

- First, assume the end to end frame transportation delay at the physical layer is fixed and is equal to 100 ms for a typical CDMA [26] channel.
- Second, assume that when a single or multiple NAKs needed to send back, the channel is always available and there is no extra queuing delay caused by other frames.

- Third, assume that a received NAK frame can always trigger frame retransmission without any queuing delay.
- Finally, assume that RLP at the receiving end can accurately predict the arrival time of a retransmitted frame so that no extra delay is needed to start the next round retrieval when necessary.
- The inter-frame time is taken as 20 ms as specified in EIA/TIA/IS-95 [26].

With the above stated assumptions, delay calculations are based on how parameters are selected for TCP and RLP layers and how the simulator is configured to advance the simulation time. First of all, each TCP segment is fragmented into a number of RLP frames, which are then transported over the physical channels of CDMA. The number of RLP frames generated is, therefore, a function of the TCP Maximum Segment Size for a given link capacity and service option. As an example, for a Maximum Segment Size of 536 bytes and one RLP frame of 19 bytes, 29 RLP frames are needed to carry one segment of full rate primary traffic. In practice, for each segment transmitted, TCP waits for the ACK until its timer expires. It is assumed that TCP timer is synchronized with the maximum number of retrievals that RLP makes to recover any frame in error. It means that RLP has to complete its maximum retrievals before the TCP time out occurs. The parameter, number of RLP retrievals, is an important factor in delay calculations. In our test bed, we have assumed the worst case scenario in which RLP delivers each packet to TCP after completing its maximum retrievals. In other words, we assumed that for each frame in error, RLP keeps the rest of the frames for the same segment until it completes its maximum retrievals and then transfers the segment to the upper layer. Hence, the Round

Trip Time (RTT) will at least be equal to the time RLP takes to complete its maximum retrials. In this way, the delay results will present the case in which each segment may have a faulty frame, which may or may not be recovered through the RLP ARQ mechanism. In any case, the segment will only be transferred to Transport layer after RLP completes its maximum retrials time. This assumption makes the calculations of Round Trip Time simpler and also generates worst case results for mean delay.

In addition to RLP introduced delay, the second factor in delay determination is TCP response to losses. If TCP finds its timer expiring without receiving an ACK, it makes an exponential back off and invokes congestion control algorithms. In exponential back off, the timer time-out value is doubled to its previous value. This also adds to the mean delay with increased number of errors in wireless channels. In other words, mean delay is expected to be higher for higher values of Frame Error Rate.

During the simulation, the transmission time for each segment is recorded as *TransmitTime* within segment object. Similarly, received time for a successfully acknowledged segment is also recorded as *ReceivedTime* in *ackObject*. The difference of these times is used by the *delayStat*, which is defined as a statistics collection object. The *StatObj* calculates Mean delay after the completion of simulation.

4.3 SIMULATOR VALIDATION

For a comparison with the results presented by [8] and to justify our simulation test bed, simulations are performed with the same parameter settings at TCP and RLP layers as used by the author. A list of selected parameters is shown in Table 4.2. For each simulation result, one of the parameters is changed from the default values specified by [11] against the fixed Frame Error Rate introduced by the physical wireless channels of

Figure 4.2: TCP Throughput if FER is applied to it directly

the CDMA environment at the RLP layer.

The results of TCP performance metrics versus FER are plotted in Fig 4.2 through Fig 4.5 for a full rate primary traffic. It is useful to mention again that the data to be

Parameter	Selected values	Default Values
Maximum Segment Size	536, 1072	536
Maximum Number of RLP Retrials	2, 3	3

Table 4.2: Selected Simulation Parameters

transferred by TCP is modeled by an infinite buffer for simulator verification.

Figure 4.2 shows a plot of TCP throughput when there is no RLP operational below it

Figure 4.3: TCP Throughput at different parameter settings

to recover the losses. In this case, the frame error rate (FER) of the physical layer is directly applied to transport layer. TCP attempts to recover the losses with its default-sliding window ARQ and congestion control algorithms. From the plot, it is clear that with the increase in FER at the wireless link the throughput of TCP is going down rapidly. For a FER of 10%, the TCP throughput drops to only 1600 bits per second.

The throughput of TCP with 2- and 3-retrials is plotted in Figure 4.3. For the two curves in the plot, a default value of TCP Maximum Segment Size is considered (536 bytes). The plot shows that 2-retrials at the RLP work fine for low Frame Error Rates but for high values of FER, the throughput of TCP goes down quickly. The performance degradation of the 2-retrial case is largely caused by an increased number of packets in error, since each packet in error causes a packet retransmission which forces congestion window to shrink to 1. At this state, TCP operates on stop and wait ARQ. The throughput is, therefore, substantially decreased. In contrast to 2- retrials, the throughput of TCP is better and nearly constant for the default value of 3- retrials at RLP. The reason for this

steady throughput is a very low probability for TCP packets to be in error while RLP is

FER (%)	2-retrial	2-MSS	default
1	7381	7587	7384
2	7378	7587	7383
3	7363	7587	7383
4	7323	7587	7382
5	7259	7587	7382
6	7156	7587	7382
7	6971	7586	7381
8	6619	7584	7379
9	6100	7581	7376
10	5770	7574	7370

Table 4.3: Comparison of TCP Throughput

making three retrials to recover the erroneous frames. As we can see from Table 4.1, the probability of a TCP packet in error is very low even for a FER rate of 10% with 3-retrials at RLP. So TCP finds itself transmitting at the maximum allowable congestion window for most of the time and the retransmission rate is very low. Hence, TCP makes the best utilization of available bandwidth. The third curve (2-MSS) in Figure 4.3 gives us a TCP response with the maximum segment Size (MSS) equal to double the default value (1072 bytes) at default (3) retrials at RLP. The throughput of TCP is found even more improved because more data bytes are now contained in a segment but the disadvantage is clear in terms of mean delay response. Table 4.3 provides a clearer comparison of the overall TCP throughput for different parameter settings in our simulator.

Figure 4.4 shows how the mean delay increases with increased channel errors. The results are plotted only for 2-retrials to show clearly the trend of the mean delay response. Figure 4.5 is a comparison of plotted mean delay with different parameter settings.

Obviously, the delay is least for 2 retrials but as we have already discussed that TCP throughput for the case of two retrials is not acceptable for high FER. For a segment size of 2MSS, the mean delay time for data transmission is very high (of the order of 4 seconds). So the advantage of having greater throughput with larger segment sizes is minimized due to larger delays introduced. This is specifically true for transmission of time sensitive services.

The results presented by [8] are shown in Fig 4.6 and Fig 4.7, which reflects a close compatibility with our simulator output. The behavior of the curve at 2-retrials can be

Figure 4.4: Mean Delay Pattern

observed to explain the close compatibility of the results with [8]. The difference in throughput at 10% FER and at 1% FER as described in [8] is $7600-6250=1350$ bits/second. The same parameter settings in the simulation test bed yields a difference of almost 1500 bits/second. This is not a big difference considering the ranges of FER. If delay statistics are taken into account, the maximum delay at 10% FER with default setting is around 2.6 seconds in our simulation test bed which is 2.5 seconds as described

in [8]. For 2-retrial, in particular, the delay obtained by our simulation test bed is 2.3 seconds which is exactly the same as reflected in [8]. However, there are two differences that can be noticed but can be explained as well. For throughput plots, Fig 4.6 shows a significant decrease in the throughput with increase in Frame Error Rate for default and 2-MSS settings. Whereas our results in Fig. 4.4 is showing nearly steady throughput against these parameter settings. We justify our analysis by explaining TCP behavior against low error probability. Both the cases mentioned are representing 3- retries at the RLP layer and the probability of a packet in error is very low for this case (refer table 4.1). Even at 10% FER, the probability that the TCP receives a packet in error is 1.36×10^{-2} .

2. TCP is expected to operate with maximum allowable window size for most of the

Figure 4.5: Delay Comparison at different parameter settings

transmission at this small error rate keeping in mind that the advertised window size is limited to 4 MSS at most. The occurrence of a packet error causes the TCP window size to shrink to 1 but since the next error is expected to happen after a long transmission (because of a very low error probability for 3-retrial), TCP attains its maximum window size rapidly after going through phases of slow start and congestion avoidance algorithms.

It continues transmitting at full window size till the occurrence of the next error. Hence, the overall throughput of the system is expected to be fairly steady for even greater FER

Figure 4.6: Throughput Results from [8]

at wireless channels when RLP maximum retrials option is set at 3.

A second difference can be noticed in the mean delay response for 2-retrials and default settings at lower frame error rates like at 1% FER. The mean delay for default setting in our results is showing larger values for lower Frame Error Rate as compared to decreased delay shown by [8]. Our assumption of worst case delay scenario is the reason for this difference. As already mentioned during the discussion of delay statistics collection that we are considering every frame loss at the physical layer has to face 3-retrials at RLP before it gets recovered or RLP fails to recover it. In other words, for every frame loss, RLP will cause a delay equal to the 3-retrial time before a packet is passes upto the transport layer. In reality, there may be few frames losses that can be recovered in 2-retrials even some of them may be recovered by a single retrial at the RLP layer. In such cases, RLP need not to cause 3-retrial delay. This assumption though caused the mean delay to to appear as approximately the same even at 1% FER but enabled us to

produce the worst scenario delay statistics with an easy implementation.

With the above discussion and comparisons, it is quite clear that our simulation results are close to those presented by [8]. The throughput and delay plots seem to follow almost the same pattern and the values are close as well. This supports the correctness of

Figure 4.7: Delay Results from [8]

our simulation test bed and consistency in its behavior with changing network conditions.

The chapter discussed various issues of our simulation test bed design. All important parameter settings to model the TCP behavior are discussed. We highlighted the assumptions made in the development of the whole simulator. We also defined and discussed the performance parameters and how they are calculated in our simulation environment. The simulation results are discussed and verified for the consistency and correctness of in behavior.

We performed all simulations till now following the assumptions made by [8] and the network parameters selected in that particular study to prove the correctness of the

simulation test bed. In his study, [8] has assumed an infinite number of packets available in the TCP buffer. This is normally not true for real traffic. The arrival of data to the TCP follows a certain pattern depending upon the traffic type. So the results obtained with such assumptions do not reflect the performance patterns of real TCP traffic. It is necessary to re-run the simulation with real incident data arrival patterns to figure out the performance of such traffic over the same protocol stack of IS-99. Moreover, the author in [8] has defined a fixed size segment (every segment is equal to 536 bytes) for the entire simulation study. Again, this is a fact that TCP does not provide same sized segmentation for data transfer. In fact, the size of the segment is a virtual quantity which is a trade off between the arrival data pattern and the allowable TCP window size at that particular instant. For example, if the data at the TCP buffer at any instant is 1300 bytes, the maximum segment size negotiated is 536 bytes and the allowable window size is 3, the complete TCP window will appear as two segments of 536 bytes each and a third segment of size 228 bytes (left bytes after two segments of maximum segments size). This scenario is not addressed in [8] while it is assumed that whenever TCP has to send data, it will have sufficient data bytes available in its buffer to fill the allowable window size with segments of maximum segment. Moreover, the maximum segment size and the maximum of RLP retries are not the only network parameters that may affect the throughput and delay performance of an Internet application. It is required to identify more network parameters that directly or indirectly effect the throughput of TCP.

With this point of view, it is worth to study the performance of Internet applications of certain traffic pattern with a true effect of TCP segmentation process. For this reason, we have developed Internet traffic generators (refer Chapter 3 for details of traffic

models). In the next chapter, we will present the analysis and performance metrics of different services using actual traffic characteristics instead of an infinite buffer of infinite number of packets. For this purpose, we will apply traffic generators, which will be generating traffic of certain characteristics. Our simulation test bed is then used to analyze and evaluate the performance metrics of those services. The test bed is slightly modified to reflect the TCP segmentation process at each data transmission.

Chapter 5

Performance of Internet Applications

5.1 PERFORMANCE PARAMETERS

In this study, we consider the Internet scenario for mobile applications. In such scenario, a typical data user might do things like read/compose short e-mail messages, respond to paging-type messages, transfer files between the mobile and base station and browsing the Internet. These Internet applications are interactive in nature. The file transfer protocol (FTP) is usually used for bulk file transfer between two machines or hosts and is delay-tolerant in nature. When the application (FTP) is used in interactive command-line mode, the control session is interactive between the client and server. For this reason, the delay that may be tolerated for this connection from the time a command is issued by the client to the time that the response from the server is received is limited. The data connection is usually delay tolerant, but is very sensitive to the effective transmission throughput. When the application (FTP) is used in the non-interactive application mode, the control session is not separated from the data transfer session. Both connections of the FTP, in this case, are delay-tolerant. Web browsing is also an interactive application that is delay-tolerant. The response time for WWW traffic can tolerate delays up to few seconds (4-5 seconds). WWW data transmission is sensitive to losses. It requires a high throughput so that contents of the downloaded document can be displayed accurately. The E-mail protocol, SMTP, is loss-sensitive and cannot tolerate any degree of errors. It requires a high data transmission throughput. On the other hand, it may tolerate a significant delay in communication.

With the above discussion of quality requirements for Internet data applications, it can be stated that all these applications may tolerate a certain amount of delay. The results and discussions in previous chapter would be enough to describe the delay performance of TCP over RLP. Plots of delay show that in even worst scenario, it is within 5 seconds limit, which is quite acceptable for delay-tolerant Internet applications. Therefore, we limit our discussions with overall throughputs of the Internet applications as “performance parameter”. Throughput provides a measure of losses that can be faced by an application in wireless environment. There are a number of parameters at TCP, RLP and network levels that can be identified as “parameters of interest” and may affect the overall TCP/RLP performance over wireless. These network parameters may include:

- Maximum number of retrials at RLP layer
- Maximum Size of a TCP Segment (MSS)
- Advertised Window Size (AWS)
- Data Size/File Size to be transmitted
- Link Capacity of the radio interface

The first two parameters were used by [8] to compute results with infinite data packets at the sender application. These parameters will now be reused for performance analysis of data application like WWW, FTP and SMTP instead of infinite data arrival.

The buffer size of the receiver entity plays an important role in determining the allowed window size of sender TCP. The receiver advertises the information of its available buffer at any instant during transmission through acknowledgement packets,

which are short, compared to data packets and also piggybacked. This information is called the advertised window size (AWS). The sender entity tunes its transmission window based on advertised window. In this way, advertised window size affects the overall TCP throughput indirectly. In [11], the range for advertised window size at any instant of transmission is specified (ranging between 2MSS and 4MSS). In addition to those specifications, larger values of advertised window sizes are also considered to highlight its effect clearly. The chosen values of advertised window for simulation study are summarized in Table 5.1.

During the validation part of TCP/RLP test bed, it was assumed that infinite data packets are available in an infinite buffer at the sender TCP. Considering actual traffic, there is a need to modify some of the assumptions. Infinite data packets would not reflect the actual traffic. The assumption of having an infinite buffer can still be considered but it is provided with the data according to the arrival traffic pattern and the file sizes. The simulated file sizes are summarized in Table 5.1 for FTP/SMTP. The file size range is chosen in a way that it can reflect small e-mail message transfers as well as the large file transfer. WWW does not define a definite file size. Instead, average document size for each arrival determines the size of the total data transmitted.

Parameter	Simulated Values
Link Capacity (Kbits/sec)	4.8, 9.6, 14.4
File/Data Size (Kbytes)	3, 30, 300, 3000
Maximum Segment Size (bytes)	536, 1072
Number of RLP Retrials	2, 3, 4
Advertised Window Size (MSS)	2, 4, 8, 32
Frame Error Rate (%)	1 to 10, 5 to 40
Traffic Type	Full Rate Primary
Internet Application	FTP/SMTP, WWW

Table 5.1: Selected Simulation parameters and their values

Finally, throughput of Internet applications is considered against the link capacity provided by the wireless channels. Wide band data services option IS-99 [11] supports a variety of link capacities but a trade off between the link capacity and the buffer capacity of the receiver end is necessary to achieve maximum utilization of the available bandwidth. Simulations are run to discuss the performance variations against the link capacity.

5.2 SIMULATION ENVIRONMENT

We obtained our results through event-driven simulations using the TCP/RLP test bed developed and discussed in the previous chapter. The following are the improvements and assumptions that are made in the TCP/RLP test bed in order to simulate Internet traffic patterns with it:

- While considering Internet traffic of type FTP/SMTP or WWW, it is assumed that the IS-99 stack will support these traffic patterns even they are not asynchronous data services. In this respect, it is worth specifying that the new standards of packet data services (IS-657, IS-707) are based on RLP

specifications of IS-99.

- Instead of an infinite buffer with infinite number of packets at sender TCP, a buffer that takes data traffic from traffic generators is considered.
- Data generation is carried out using the traffic generators to produce Internet traffic of desired pattern. The models for these generators have been discussed in chapter 3.
- For the simulator validation part of the previous chapter, TCP segments, produced from infinite buffer, have fixed size (maximum segment size MSS). In practice, this is not the case for real traffic. TCP segments may have variable sizes within the range of maximum segment size (MSS) depending on the amount of data waiting to be transmitted and the TCP allowed window size. TCP usually attempts to break the available data into segments of size equal to maximum window size (MSS) but if the allowed window size is not a multiple of MSS, it may transfer a segment with a different suitable size. For this reason, the simulator is modified to allow variable segment sizes that are generated by sender TCP according to its limitations.
- Only data sessions are considered for WWW and FTP data transfer. When a data session ends, TCP closes the connection. A new connection/session needs to be opened for further transmission.
- Connection setup and tear down procedures and delays are not considered.

5.3 RESULTS AND ANALYSIS

Simulations are performed with WWW and FTP/SMTP traffic patterns over TCP/RLP test bed. We define normalized throughput as the ratio of achieved throughput to the wireless link capacity. The simulation results are presented keeping TCP normalized throughput as the main performance parameter and analyzing the effects if one of the network parameters is varied. These parameters have already been identified, therefore, the results are categorized by varying exactly one parameter each time. In addition, initially all the parameters are kept within the range specified in [11] but the buffering capacity of the receiver end is discussed for larger values to show the enhancement in overall throughput.

5.3.1 Performance of FTP/SMTP

The performance of File Transfer applications is discussed in the following section. A traffic generator is used to provide a FTP/SMTP traffic pattern to the TCP/RLP simulator. The generator model has been discussed in chapter 3.

5.3.1.1 Effect of Number of RLP Retrials

The effect of maximum number of RLP retrials for File transfer application is plotted in Figure 5.1. The numbers of RLP retrials selected values are 2 and 3 retrials. The channel conditions are modeled by running simulations for a range of frame error rates (FER). The simulated frame error rates vary from 1% up to 10%.

For smaller frame error rates, throughput is nearly the same for 2 and 3 retries as reflected in figure 5.1. When frame error rate (FER) is increased beyond 6%, the curve for normalized throughput at 2-retries is showing a decrease from 0.59 (at 6%) to 0.57 (at 10%) while it is nearly constant for 3-retries from 1% FER to 10%. This is due to the fact that if number of retries is increased at RLP, the probability of TCP packets to be received correctly increases. In other words, the probability of TCP packets to be received

Figure 5.1: Effect of 2, 3 RLP retries

in error is decreased. This prevents the TCP to make retransmissions and to cut down the transmission window. The over all effect is the increased throughput.

The frame error rate (FER) of the wireless channels may temporarily increase significantly when a mobile passes through a long coverage hole such as tunnel. For this reason, the effect of maximum number of RLP retries is also plotted for larger values of frame error rate and larger settings of the maximum number of RLP retries made to recover lost packets at its level. Figure 5.2 shows the throughput results against frame

error rates ranging between 5% and 40%.

The normalized throughput for 2-retrials case is dropping significantly for frame error rate above 15% while 3-retrials seem to be recovering most of the losses up to 25% of FER. A nearly constant throughput is maintained by 4-retrials even for a very lousy environment of 35% FER.

It is clear from the above discussions and plots that the default value of 3-retrials seems to recover losses even for higher frame errors but there is a delay cost (as discussed in previous chapter). It can also be stated that the results for a 2-retrial setting at the RLP are well acceptable keeping in mind the fact that CDMA environments are build to

Figure 5.2: Effect of RLP retrials for Burst of Errors

provide low frame error rates (up to 1 %) for most of the transmission. In this scenario, a retransmissions setting of 2-retrials is enough to recover most of the losses and very few errors are left for reliable transport layer to recover. Hence, for the rest of the analysis, 2-retrials at the RLP layer will be considered as the default value.

5.3.1.2 Effect of Maximum Segment Size

The effect of maximum segment size on file transfer protocol FTP is plotted in figure

Figure 5.3: Bar Representation of Maximum Segment Size Effect

FER (%)	MSS=1072 (bytes)	MSS=536 (bytes)
1	5863	5719
2	5863	5718
3	5861	5716
4	5860	5714
5	5857	5707
6	5857	5717
7	5856	5690
8	5855	5674
9	5850	5648
10	5846	5506

Link Speed=9.6kbits/sec

AWS=4,

number of RLP retrials = 2

File Size = 3000 kbytes

Table 5.2: Effect of Maximum Segment Size on Throughput

5.3 and also summarized in Table 5.2. The segment size can be varied between 536 bytes (default in IS-99) and 2072 bytes. The simulated values of maximum segment size are the default value (536 bytes) and a segment of size double than the default value. Hence, the results show the effect of 536 bytes (1 MSS) and 1072 bytes (2MSS) on throughput performance. Clearly, with bigger segments, the performance seems to be better but again the cost is larger delays.

In the previous chapter, it has been shown that if segment size is increased from 536 to 1072, the one way propagation delay is increased from 2.5 seconds to almost 4 seconds whereas the gain in throughput is not very high. The same pattern followed here with FTP and throughput is increased from 5506 bits/sec to 5846 bits/sec at 10% FER by doubling the segment size but of course it will cause greater delays.

To limit the delay, the default value of maximum segment size (536 bytes) will be considered for the rest of analysis and simulations.

5.3.1.3 Effect of Advertised Window Size (AWS)

According to TIA/EIA/IS-99, the advertised TCP window size should be 2 to 4 times of the maximum Segment Size (MSS). For this reason, all previous analysis was carried out with Advertised Window Size (AWS) of 4 times the MSS keeping it within specifications. In this section, it is assumed that the receiver can advertise more buffering capacities, and the effect of this increased buffer is simulated to show the performance enhancements. The values selected for the receiver's buffer are based on buffering capabilities in recent networks which shows that receivers can advertise a window size of even 32 MSS. Also, it is desired to discuss the effects of larger buffers at the receiver end.

The simulation results with varied Advertised Window Sizes are summarized in Table 5.3 and also plotted as a bar graph in figure 5.4. It is evident from the results that by just doubling the receiver's capacity from 4 to 8 MSS, the throughput of file transfer is increased from 5719 bits/sec to 6863 bits per sec (for a link capacity of 9.6 bits/sec) at 1% FER. At 10 % FER, this difference is again noticeable where the increase in throughput is almost 100 bits per sec. The most noticeable gain is in normalized throughput or link

TCP Throuhput (bits/sec)

FER (%)	AWS=2	AWS=4	AWS=8	AWS=32
1	4288	5719	6863	7234
2	4288	5718	6862	7232
3	4289	5716	6861	7231
4	4288	5714	6853	7215
5	4293	5707	6827	7201
6	4280	5717	6818	7182
7	4293	5690	6810	7151
8	4288	5674	6784	7106
9	4293	5648	6703	6986
10	4211	5506	6488	6917

Link capacity=9.6 k bits/sec

number of RLP retrials =2

MSS=536

file size=3Mbyte

Table 5.3: Effect of increasing Receiver's Buffer

utilization of the file transfer protocol. In figure 5.4, an increase of 4 to 8 mss in advertised window size has caused the link utilization to increase from 59% to 72%. This can further be increased to 76% if the buffering capacity at the receiver is enhanced up to 32 MSS. These results clearly reflect the importance of buffering provisions at the mobile station.

Figure 5.4: Effect of Advertised Window Size

In a wireless environment and considering a data transfer from base station to mobile station, the buffering capacity of mobile stations cannot be expected to be very large but the effect of greater advertised windows should be considered while designing any mobile system.

5.3.1.4 Effect of File/Data Size

In the previous chapter, the simulation results were described for an infinite buffer containing infinite number of data packets to be transmitted. In practice, this is not true and the sender buffer should be receiving data according to arrival pattern. For the transfer protocol (FTP), the arrival pattern is specified by [17]. The size of the file to be transferred is an important parameter in determining the performance and utilization of the network resources. Particularly, the link capacity together with file size can be used to determine the optimum advertised buffer required at the receiver for each transmission window.

FER (%)	3 kbytes	30 kbytes	300 kbytes	3000 kbytes
1	5170	6686	6844	6861
2	5170	6686	6844	6861
3	5170	6686	6844	6861
4	5170	6686	6844	6861
5	5170	6686	6844	6854
6	5170	6686	6809	6857
7	5170	6686	6809	6850
8	5170	6686	6844	6848
9	5170	6686	6809	6842
10	5170	6686	6741	6702

AWS=8 MSS

link speed= 9.6kbits/sec

MSS=536 bytes

number of RLP retrials =2

Table 5.4: Effect of File Size on Throughput

Table 5.4 provides with the trend of overall throughput improvement as larger files

Figure 5.5: Effect of File Size on Throughput

are transferred over the link. For smaller files, the throughput is constant irrespective of the Frame Error Rate (FER) but there is a significant difference in throughput achieved for 3 Mbytes file transfer for different channel conditions (at 1% FER and at 10% FER). Figure 5.5 represents this information in the form of a bar graph.

The file size and receiver's advertised buffer capacity also has trade off for the same

Figure 5.6: Combined Effect of AWS and File Size

link capacity. If the advertised window is smaller, even if the file to be transferred is big,

Figure 5.7: Bar Representation of AWS Effect with File Size

the overall throughputs do not change appreciably.

Figure 5.6 and 5.7 shows the variation and trade offs between file size and advertised window size for the same link capacity. It can be observed that a smaller file (3 Kbytes)

makes a considerable change in throughput just by doubling the receiver's buffer from 2 MSS to 4 MSS. But if the advertised window is further increased, it will not make any useful effect on throughput obtained by transferring a 3 Kbytes file (figure 5.6). In contrast to this, a larger file of 3 Mbytes has nearly the same throughput for smaller advertised window (because it makes the maximum use of available capacity and window size with its larger arrival of data). But this increases with increased receiver's capacity (more data can be in flow with larger available window size). The tradeoff between File Size and advertised window is also important in determining the sender's buffer size which is assumed to be infinite throughout the study but can be traded off accordingly.

5.3.1.5 Effect of link capacity

According to EIA/TIA/IS-99, the standard supports a variety of link capacities ranging from 4.8 Kbits/sec to 14.4 Kbits/sec for both primary and secondary traffic. The link capacity also affects the quality of file transfer. For simulation purposes, the number

Link capacity	bits per frame	number of RLP frames
4.8	96	45
9.6	192	29
14.4	288	15

Table 5.5: Relationship between Link Capacity and Number of RLP frames

of RLP frames that can be derived from one TCP segment is different for each link capacity. For 9.6 kbits/sec, the simulations were carried out with 29 frames out of one TCP segment of size 536 bytes. The link capacities of 4.8 kbits/sec and 14.4 kbits/sec have more number of bits in the same frame duration. The number of bits for each link

capacity and number of RLP frames that can be derived from one TCP segment of size 536 bytes are summarized in Table 5.5.

Simulations are run for different link capacities against different Frame Error Rates. The results are summarized in Table 5.6 and plotted as a bar graph in Figure 5.8. The frame error rate still making a difference where the results are showing differences in throughput for different channel conditions (1% FER and 10% FER). In fact, if we consider Table 5.6, it is clear that we have higher throughput at higher link capacities. For example, the throughput is 4206 kb/sec, 6488 kb/sec and 10211 kb/sec at 4.8, 9.6 and 14.4 kb/sec link capacities respectively (at 10% FER). This is showing a clear increase in

FER (%)	4.8 kb/s	9.6 kb/s	14.4 kb/s
1	4290	6863	11435
2	4290	6862	11435
3	4290	6861	11423
4	4293	6853	11411
5	4282	6827	11339
6	4276	6818	11324
7	4277	6810	11360
8	4264	6784	11322
9	4229	6703	11031
10	4206	6488	10211

AWS=8

number of RLP retrial=2

MSS=536

file size=3000k bytes

**Table 5.6: Effect of Link Capacity on Throughput
(bits/sec)**

throughput with the increase in available bandwidth in a wireless environment. But if the normalized throughput or link utilization is considered, we observe what we get in figure 5.8. The lowest bandwidth wireless environment best utilizes the link capacity. This is due to the fact that the File Size of 3 Mbytes proved to be more than enough to keep the

link (4.8 Kbits/sec) busy all the time where the buffer advertised capacity was enough large as well (8 times the Maximum Segment Size (MSS).)

This trend does not follow for 9.6 kb/sec and 14.4 kb/sec. Infact, 9.6kb/sec has the worst link utilization curve. The reason for this strange behavior can be explained considering the varying effects of framing and data arrival patterns for different link capacities. It has already been stated that the frame sizes are different for different link capacity (see table 5.5). In other words, when comes to frmaing, a TCP segment may produce different number of frames for 4.8, 9.6 and 14.4 kb/sec. Now, the arrival data rate is same for all available bandwidth in the simulation setup, the maximum segment size MSS is also constant but the frame sizes are different and hence different number of frames are generated with varying bandwidth. When the error model is applied to the incident frames during simulation (which produces an error repeatedly making a count of arriving frames), errors produced are different for different bandwidth. In other words, the frames in error are not the same for all link capacities, the packets in error left unrecovered are not the same and hence the TCP reaction to all available errors is not the same for all link capacities. This chain reaction caused initially by the framing has produced the normalized throughput at 9.6 kb/sec to be worst. This means that the timing of framing and error production has caused the normalized throughput go down.

Figure 5.8: Effect of Link Capacity

As mentioned above, the throughput, in addition to link capacity depends both on file

Figure 5.9: Effect of Link Capacity

size and advertised window size. To analyze the effect, simulation results are plotted for throughput with different combinations of file sizes and advertised window sizes in

Figure 5.10: Combined Effect of AWS and Link Capacity

Figure 5.9 through 5.10.

The combined effect of file size and link capacity is clear from Figure 5.9. For link capacity of 14.4 Kbits/sec, the throughput is only 0.53 if the transmitted file size is small (3 Kbyte) while it is 0.88 for a large File Size.

The advertised window size has already been discussed for its effect on performance. In figure 5.10, it is combined with link capacity to show the over all effect. For smaller receiver's buffer, throughput is low for high link capacity if the being transferred is of same size. In this case, the sender has to buffer most of the data with it because the transmission window does not allow transmitting larger data in one window transmission. If the buffer at receiver is large enough, the transmission window will increase thereby increasing the over all throughput of the file transfer.

5.3.2 Performance of WWW

WWW document arrival patterns are used with the TCP/RLP simulator to analyze the performance of Web traffic. These traffic patterns are generated through a WWW traffic generator. The generator model is taken from [14] and has already been discussed in chapter 3. Simulations were run considering a web user that spends 2 hours on average doing web browsing. Moreover, Only data sessions are considered i.e., a session is terminated by TCP during user thinking time and a new session is opened for the next document arrival. This is how TCP operates in practice.

5.3.2.1 Effect of Maximum number of RLP retrials

Simulations were run for 2- and 3-retrials against Frame Error Rate increasing from

FER (%)	2-retrial	3-retrial
1	4057	4061
2	4057	4061
3	4057	4061
4	4056	4061
5	4057	4061
6	4048	4061
7	4057	4061
8	4026	4058
9	4007	4052
10	3874	4050

*AWS=4
 MSS=536 bytes
 Avearge Document Size=250 bytes
 Link Capacity=9.6 kbit/sec*

Table 5.7: Effect of RLP retrials

1% to 10%. The results are plotted in Figure 5.11 showing Normalized TCP throughput against Frame Error Rate for 2- and 3-retrials at the RLP layer. The throughput for 2-retrials is showing few variations. Though the difference does not look very big as it only

Figure 5.11: Comparison of 2, 3 RLP retrievals

varies between 0.418 and 0.423 for 10% to 1% FER. The throughput at 3-retrials is fairly steady. An important observation is the comparison with FTP. The throughput of WWW traffic is quite low as compared to what achieved in case of File Transfer Protocol. It is clearer from Table 5.7 that the maximum achievable throughput for a link capacity of 9.6 k bits/sec is 4.061-k bits/sec, while it was about 7.6 kbits/sec for FTP for the same Link Capacity. The difference is created by the document pattern and document size. The FTP model shows that the application generates Constant Bit Rate (CBR) traffic patterns while WWW document pattern is random in nature and follows certain distribution. So the total data size is not that much large as a complete File Size in case of FTP.

The effect of large burst of errors is also plotted in Figure 5.12 to show the performance variations against 2-, 3- and 4-retrials at the RLP layer. The pattern is the same as in case of FTP, i.e., for burst of errors, 4-retrials at the RLP result in best throughput at higher error rates. The performance at 3-retrials is very close to 4-retrials up to 25% FER.

The reason for low TCP throughput in case of WWW traffic is the amount of data being generated by WWW arrivals. This can better be explained when the effect of data size will be considered.

Finally, it may be stated that most of the errors are being recovered by 2-retrials for

Figure 5.12: Effect of 2, 3, 4 RLP retrials

FER up to 6% so a selection of 2-retrials at the RLP layer as a default value is being considered for the rest of the analysis

5.3.2.2 Effect of Maximum Segment Size

Variations in WWW performance with two different Maximum Segment Sizes (MSS) are plotted in Figure 5.13 and also presented in Table 5.8. The segment size is not really making a considerable improvement in throughput. At 10% FER, the difference is 80 bits/sec by doubling the segment size. For low FER, this difference is increased to 136 bits/sec, which is again not very large. This advantage is obtained at the cost of increased one way delay experienced by the segments. With these discussions, Maximum Segment

Size of 536 bytes will be considered as default segment size for the rest of the analysis.

FER (%)	536 bytes	1072 bytes
1	4057	4195
2	4057	4195
3	4057	4185
4	4056	4183
5	4057	4156
6	4048	4147
7	4057	4118
8	4026	4099
9	4007	4092
10	3874	4089

AWS=4MSS
Avg. Doc. Size=250 bytes
Link Capacity=9.6 kb/sec
Number of RLP retrials=2

Table 5.8: Maximum Segment Size variations

Figure 5.13: Effect of Maximum Segment Size

5.3.2.3 Effect of Advertised Window Size

The advertised window size (AWS) reflects the capacity of the receiver's buffer. For larger advertised buffer, more data can be in flow. The variations in WWW performance at variable channel conditions and different receiver buffer (AWS) are plotted in Figure 5.14.

The results for buffer capacity are quite interesting as can be seen from Table 5.9. Throughput of WWW is increasing with a increase of window advertisement up to AWS=8MSS. The trend is quite normal till here but a further increase in Advertised

Figure 5.14: Effect of Advertised Window Size

Window Size does not cause any noticeable improvement in throughput performance. This is because of the fact that the size of AWS=8 MSS is enough to transmit a document arrival of 250 bytes/arrival. That is why, an increase in the receiver's advertised buffer did not result in any performance improvement.

FER (%)	AWS=2MSS	AWS=4MSS	AWS=8MSS	AWS=32MSS
1	3086	4057	4452	4443
2	3084	4057	4451	4441
3	3083	4057	4449	4441
4	3081	4056	4448	4439
5	3082	4057	4445	4436
6	3071	4048	4440	4430
7	3076	4057	4433	4426
8	3068	4026	4415	4405
9	3067	4007	4395	4388
10	2995	3874	4408	4371

MSS=536 bytes

Avg. Doc. Size=250 bytes

Number of RLP retrials=2

Link Capacity=9.6 kb/sec

Table 5.9: Effect of Advertised Window Size

5.3.2.4 Effect of Link Capacity

Figure 5.15 shows the effect of Link Capacity on throughput of Web traffic.

Figure 5.15: Effect of Link Capacity

It can be observed that WWW performance is almost unchanged for 9.6 kb/sec and 14.4 kb/sec. At 4.8 kb/sec, the throughput is even lower than higher data rates. Table 5.10

FER (%)	4.8 kb/sec	9.6 kb/sec	14.4 kb/sec
1	3643	4452	4456
2	3641	4451	4456
3	3641	4449	4456
4	3636	4448	4454
5	3641	4445	4451
6	3613	4440	4446
7	3613	4433	4440
8	3613	4415	4425
9	3545	4395	4416
10	3378	4408	4397

AWS=8 MSS

MSS=536 bytes

Avg. Doc. Size=250 bytes

Number of RLP Retrials=2

Table 5.10: WWW Throughput variations for different Link Capacity summarizes the results in bits/sec against channel errors. These results are quite expected and self-explanatory.

5.3.2.5 Effect of Data Size

While discussing the effects of various parameters in previous sections, it has been stated repeatedly that the degraded performance of WWW traffic over the simulation setup is mainly caused by the smaller data bytes being generated. The average document size for each WWW arrival has been considered as 250 bytes per arrival [14]. To show the effect of larger document arrivals, simulation results are plotted in Figure 5.16. The average document size is varied between 250 and 750 bytes per arrival.

Figure 5.16: Effect of Variations in Avg. Doc. size

Table 5.11 provides another view of WWW performance against varying Frame Error Rates and different document sizes. Though the performance is improved for larger data

FER (%)	250 bytes	500 bytes	750 bytes
1	4448	6008	6010
2	4446	6006	6008
3	4442	6006	6006
4	4432	6004	6000
5	4417	6000	5993
6	4402	6004	6006
7	4386	5956	5957
8	4367	5910	5911
9	4318	5855	5856
10	4295	5672	5672

AWS=4 MSS
 MSS=536 bytes
 Link Capacity=14.4 kb/sec
 Number Of RLP retrials=2

Table 5.11: WWW performance against Data Size

arrivals, if the comparison is made only with reference to data size, still the maximum

throughput which was attained is not high enough. It can be seen that for 750 bytes of data, a 14.4 kbits per second link is resulting in a throughput of 6.01 kbits/sec. There is still one parameter that is creating this problem. The Advertised Window Size in determination of the throughput dependency on Data Size was taken as specified in [11] i.e., $AWS=4 \text{ MSS}$. Hence, even if there are more data bytes waiting at the base station, the overall throughput of WWW is not improving considerably. To analyze the effect of Advertised Window Size combined with Average Document Size, simulation results are

Figure 5.17: Combined Effect of Doc. Size and AWS

plotted in Figure 5.17.

It can be seen from Figure 5.17 that performance of WWW is increased substantially for Average Document Size of 750 bytes when it is combined with an Advertised Window Size of 32 MSS. This again reflects the importance of the receiver's buffer size for a better performance. These results are taken with Link Capacity of 14.4 k bits/sec. Table 5.12 is another view of WWW throughput dependency on document size and

receiver's buffer.

Doc. Size (bytes)	AWS=2	AWS=4	AWS=8	AWS=32
250	4048	4396	4396	4386
500	4053	5672	7025	8755
750	4053	5672	7029	10406

MSS=536 bytes

Link Capacity=14.4 kb/sec

Number of RLP retrials =2

Table 5.12: Doc. Size and AWS Effect on Throughput

Another aspect of receiver's buffer analysis is its impact on sender's buffer. If the data to be transmitted is large and the Advertised Window Size (AWS) does not allow an efficient data flow between the two entities, the sender will have to maintain a large queue of arriving data bytes. If the sender is given a limitation of buffer size, the data that will arrive when the buffer has already been filled, has to be dropped. Hence, a tradeoff between sender and receiver buffer sizes based on the data size should be made.

Chapter 6

Conclusions and Future Work

6.1 CONCLUSION

This thesis is based on the problem of poor Internet Protocol performance in wireless environments. The thesis presented TCP operation for fixed networks and highlighted the causes that degrade the performance of same Internet protocol in wireless losses. The problem stems from the fact that the Transport Control Protocol has been designed for fixed networks and it was never considered that the same protocol may be used in mobile communications.

We carried out a study of the currently available protocols for improving TCP over wireless channels. Advantages and limitations of each approach were discussed in detail. Of the surveyed proposals, we picked the Radio Link Protocol (RLP) defined in IS-99 stack for simulation with Internet traffic and evaluated the performance parameters. In order to achieve our objective, we built a TCP/RLP test bed using MODSIM III (OO language) . The validation of the test bed was also included along with the design discussions. The test bed is generic and can be used with different traffic models and different network conditions.

To perform the performance evaluation of Internet traffic, WWW and FTP/SMTP traffic generators were also built. We discussed the models of these generators as well.

Integrating the traffic generators with TCP/RLP test bed, we carried out a comprehensive performance study for WWW and FTP/SMTP applications. For this

purpose, we first discussed the performance requirements for these applications and defined the performance parameters. The normalized throughput was chosen as the main performance parameter. A number of network parameters were identified which may affect the throughput of the Internet applications. The effects of the maximum number of RLP retrials, the maximum segment size at TCP layer, the size of advertised window, the data size, the bit rate of the radio interface are computed and discussed for each traffic type. We presented a number of plots and tables to support the discussions.

6.2 SUMMARY OF RESULTS

As mentioned above, effects of a number of parameters were examined for WWW and FTP/SMTP traffic types. We attempted to vary the network parameters within the specifications of the IS-99 standard but also presented some results with different values to show the performance improvements with increased buffer provisioning at the receiver and to suggest few research options.

It has been observed that most of the frame losses were recovered with 2-retrials at the RLP layer. A difference of few hundred bites per second could be achieved for Frame Error Rate between 6% to 10 % but that is not significant. We also presented a performance comparison for 2-, 3-, 4-retrials at the RLP layer to show their effect against a burst of errors. The Frame Error Rate, for this purpose, was increased up to 40%. It was observed that 4-retrials maintains the throughput till 35 % FER but the performance of 3-retrials is also comparable up to 30% FER.

Larger segments at the TCP layer can put more data in flight for a single transmission. That is why, doubling the Maximum Segment Size results in overall throughput

performance improvement. On the other hand, the increase in segment size also generates longer delays. So the advantage of improved throughput is minimized. For this reason, we considered the default segment size of 536 bytes for the rest of analysis.

An optimum combination of Advertised Window Size, Link Capacity and Data size results in good system throughput. To show the effects of these network parameters, we presented results with individual parameter variation and their effect on throughput. We also introduced a combination of different settings and discussed the overall effect. An important observation was made with the variation in Advertised Window Size coupled with Data size. The size of the receiver's buffer is limited to 2 to 4 times the Maximum Segment Size in the IS-99 specifications. For small data transfer like small e-mail messages (3 Kbytes), the size for Advertised Window specified seemed enough but large file transfers (300 to 3000 Kbytes) may create long queues at the sender (Base Station in our case) with less link utilization. We simulated large file transfers with larger receiver's capacity advertised (8 MSS and 32MSS) to show the comparative improvement in throughput at the same link capacity of the radio interface. The results show the importance of buffer provisioning at the receiver entity (Mobile Terminal) in our case. This also prevents longer queues at the Base Station since most of the data arriving can be transmitted with larger allowed window size at the sender application.

6.3 FUTURE RESEARCH AREAS

In this comparative study, we tried to cover the important issues related to performance improvement for Internet traffic. However, some issues, which need further research, are discussed as follows:

- We performed Performance evaluation of WWW and FTP/SMTP traffic types. These Internet applications are loss-sensitive but delay-tolerant. The Radio Link Protocol introduces delays while recovering losses due to errors. A study of the performance of time-sensitive applications can be a good research area.
- We examined the effects of Advertised Window Size and Data Size for a given Link Capacity. It was observed that the limited receiver's buffer (as specified in IS-99) for transfer of large files results in less data in flight and the sender TCP has to buffer a large amount of data. There should be some trade off between the buffer of the two entities according to the data arrival. Our results and discussions may help in any research in this respect.
- We carried out the study with the primary traffic option of Radio Link Protocol. This leaves an option to study the performance of integrating different Internet applications with the Radio Link Protocol.
- Performance of Multimedia applications and their scheduling issues with the radio Link Protocol may be investigated.
- We have presented the results with a data flow from Base Station to mobile terminal on a forward CDMA channel. In this scenario, the number of users supported and their contention for network resources was not an issue. The performance study on the reverse CDMA channel taking system capacity in consideration is an open research area.
- The physical channel conditions combined with the Radio Link Protocol can

be studied for their effects. In this regard, effects of power control mechanisms, coding and interleaving at the channel level may improve performance.

References

- [1] H.Balakrishnan, V.N.Padmanabhan, S.Seshan and R.h.Katz; "A comparison of mechanisms for improving TCP performance over wireless links"; Computer Science Division, Department of EECS, University of California at Berkeley available at: <http://tcpsat.lerc.nasa.gov/tcpsat/papers.html>.
- [2] A.Barke and B.R.Badrinath; "I-TCP: Indirect TCP for Mobile Hosts, Technical Report DCS-TR-314"; Rutgers University, Oct 1994. available at: <http://tcpsat.lerc.nasa.gov/tcpsat/papers.html>
- [3] E.Amir, H.Balakrishnan, R.H.Katz; "Efficient TCP over Networks with Wireless Links; University of California, Berkeley available at: <http://tcpsat.lerc.nasa.gov/tcpsat/papers.html>
- [4] R.Caceres and L.Iftode; "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments"; *IEEE JSAC special issue on Mobile Computing Networks*, Vol. 13, no. 5, June 1995.
- [5] S.Bannerjee, J.R.Goteti, G.Krishnamoorthy; "Extending TCP for Wireless Networks"; Department of Computer Science, University of Berkeley.
- [6] M.Stangel, V.Bhargavan; "Improving TCP Performance in Mobile Computing Environments"; Proceedings of VTC, 1998.
- [7] K.Brown, S.Singh; "M-TCP: TCP for Mobile Cellular Networks"; Department of Computer Science, University of South Carolina, Columbia, July, 1997.
- [8] Gang. Bao; "Performance Evaluation of TCP/RLP Protocol Stack over CDMA Wireless link"; *Wireless Networks*, no. 2, 1996, pp 229-237.
- [9] Douglas.E.Comer; "Interworking with TCP/IP, Vol. 1, principles, protocols and Architecture, Prentice Hall, 3rd edition, 1995.
- [10] A. Ballardie; "RFC 1949: Scalable Multicast Key Distribution"; May 1996 available at: <http://www.cis.ohio-state.edu/htbin/rfc/INDEX.rfc.html>
- [11] EIA/TIA/IS-99; "Data Services option Standard for Wideband Spread Spectrum Digital Cellular Systems, 1995.
- [12] W.Richard Stevens; "RFC 2001, TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery Algorithms", January 1997 available at : <http://sunsite.auc.dk/RFC/rfc/rfc2001..>
- [13] V.Jacobson; "Congestion Avoidance and Control"; *Computer Communications Review*, Vol. 18, no. 4, Aug. 1998, pp 314-329.
- [14] Shuang Deng.; "Empirical Model of WWW Document Arrival at Access link"; ICC, June 1996, Dallas, Texas, pp 1797-1802.
- [15] B.A.Mah; "An Empirical Model of HTTP Network Traffic"; IEEE INFOCOM'97, April.1997

- [16] Mark.E.Crovella, A.Bestavors; *"Self Similarity in WWW Traffic: Evidence and Possible Causes"*; ACM SIGMETRICS International Conference, May 1996, pp 160-169.
- [17] ETSI Standard for UMTS, TR 101, 1998.
- [18] R.Yavatkar, N.Bhagwat; *"Improving end-to-end Performance of TCP over Mobile Internetworks"*; Workshop on Mobile Computing Systems and Applications, December 1994.
- [19] K.Fall, S.Floyd, " Simulation based comparison of Tahoe, Reno and SACK TCP", Computer Communication Review, July 1996 available at: <http://tcpsat.lerc.nasa.gov/tcpsat/papers.html>
- [20] H.Balakrishnan, S.Seshan, R.Katz; *"Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks"*; ACM Wireless Networks, December 1995 available at: <http://tcpsat.lerc.nasa.gov/tcpsat/papers.html>
- [21] A.Myles, D.Skellern; *"Comparison of Mobile Host Protocols for TCP"*; Technical Report, Macquarie University in Sydney (Australia), April 1993.
- [22] A.DeSimone, M.Chuah, O.Yue; *"Throughput Performance of Transport-layer Protocols over Wireless LANs"*; Proceedings Globecom' 93, December 1993.
- [23] A.Chan, D.Tsang, S.Gupta; *"TCP (Transport Control Protocol) over Wireless Links"*; 47th, VTC'97, pp 1326-1330.
- [24] P.Bhagwat, P.Balacharya, A.Krishna, S.Tripathi; *"Using Channel Rate State Dependent Packet Scheduling to Improve TCP Throughput over Wireless LANs"*; Wireless networks, no. 3, 1997, pp91-102.
- [25] *"RFC 793: Transport Control Protocol"*; by Information Science Institute, University of Southern California, 1981.available at : <http://sunsite.auc.dk/RFC/rfc/rfc793.html>
- [26] EIA/TIA/IS-95 available at: <http://47.161.113.52/syseng/2H00/was/>
- [27] EIA/TIA/IS-657; *"Data Services option Standard for Wideband Spread Spectrum Digital Cellular Systems"* available at: <http://47.161.113.52/syseng/2H00/was/>
- [28] EIA/TIA/IS-707; *"Data Services option Standard for Wideband Spread Spectrum Digital Cellular Systems"* available at: <http://47.161.113.52/syseng/2H00/was/>
- [29] S.Floyd; *"TCP and successive Fast Retransmits"*; Technical Report, Lawrence, Berkeley Laboratory, May 1995.
- [30] R.Yavatkar, N.Bhagwat; *"Improving end-to-end Performance of TCP over Mobile Internetworks"*; Workshop on Mobile Computing Systems and Applications, December 1994.
- [31] W.R.Stevens; *"TCP/IP Illustrated, Vol. 1, Addison Wesley, 1994"*.
- [32] Gary.R.wright, W.R.Stevens; *"TCP/IP Illustrated, Vol. 2: The Implementation, Addison Wesley, 1994"*.
- [33] A.Bhushan; *"RFC354: File Transfer Protocol (FTP", August, 1972 available at <http://cs.ru.ac.za/func/RFC/rfc/rfc354.html>*
- [34] J.Nagle; *"RFC896: Congestion Control in IP/TCP Internetworks"*; January 1984 available at :

- <http://www.cis.ohio-state.edu/htbin/rfc/rfc896.html>.
- [35] V.Jacobson, L.Braden , "RFC 1072: TCP Extensions for Long-Delay Paths", October 1988 available at : <http://sunsite.auc.dk/RFC/rfc/rfc1072.html>.
 - [36] M.Mathis, J.Mahdavi, S.Floyd, A.Romanow, "RFC 2018: TCP Selective Acknowledgement Options", October 1996 available at: <http://sunsite.auc.dk/RFC/rfc/rfc2018>.
 - [37] R. Fileding, J. Gettys, J Mogul, H.Frystyk, T.Berners, "RFC 2068: Hypertext Transfer Protocol", January 1997 available at: <http://sunsite.auc.dk/RFC/rfc/rfc2068>.
 - [38] J. Postel, " RFC 821: Simple Mail Transfer Protocol", August 1982 available at <http://www.cis.ohio-state.edu/htbin/rfc/INDEX.rfc.html>
 - [39] H.M.Syed, K.Das, M. Devetsikiotis," TCP performance and buffer provisioning for Internet in wireless networks", MASCOT'99, October 1999.
 - [40] H.M.Syed, K.Das, M.Devetsikiotis, " TCP performance over wireless and effect of network parameters", CCB'99, Ottawa, November 1999.