Performance of TCP over Wireless Links

A Thesis Submitted to the College of

Graduate Studies and Research

in Partial Fulfillment of the Requirements

For the Degree of Master of Science

in the Department of Computer Science

University of Saskatchewan

Saskatoon, Saskatchewan

by

Venkata K. Josyula

April 2000

Permission To Use

Abstract

As wireless links work their way into the Internet fabric, it becomes increasingly important to understand how to support the additional requirements presented by wireless connections and how to accommodate the different characteristics of these connections. As a protocol designed for wired networks, the Transmission Control Protocol (TCP) operates under some basic assumptions that may not be true for wireless links. Being the foundation of many applications on the Internet, TCP must perform well in the wireless environment if wireless links are to be integrated seamlessly.

The goal of this work is to study the behaviour of various TCP mechanisms in the presence of wireless links and the effect of these mechanisms on end-to-end connection performance. The research was conducted in two phases. The first phase focuses on a measurement study of TCP performance in an experimental wireless testbed. This helped to identify the problems encountered by TCP in a wireless environment. The effect of loss on the behaviour of various TCP mechanisms is examined.

The second phase consists of a simulation study to explore possible solutions to problems encountered by TCP in a wireless environment. Several "proxy mechanisms" have been proposed to address these problems. The performance of one particular proxy model was evaluated using simulations and it was found that the proxy mechanism significantly improves the performance of a TCP connection for low and moderate error rates. For higher loss rates on the wireless link, the proxy mechanism could not completely shield the sender from the effects of loss on the wireless link.

Acknowledgements

I would like to profoundly thank my supervisor Dr. Rick Bunt for his guidance and support. I also appreciate his patience during the long time I was away. I am also thankful to my thesis committee members Dr. Carl McCrosky, Dr. Herb Yang and Dr. Eric Salt for their insightful comments and valuable feedback on this work. I am grateful to TRLabs for providing me with the necessary equipment for the wireless experimental testbed and for the financial support during the earlier part of this work.

I would like to thank Greg Oster to putting up with my requests for file space for the trace files and quickly responding to all my problems with remote access. Many thanks to Maureen Desjardins and Jan Thompson for helping me with the admin related stuff.

My parents and my cousins were a constant source of encouragement for me and always believed that I would finish my thesis someday. I would also like to greatly thank my friends in Saskatoon and the US for all their help, and their constant support. I can never forget the long conversations that helped me come back to defend my thesis after I almost gave up.

I would like to dedicate this work to my parents.

Table Of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The need to access and disseminate information has resulted in an explosive growth of data networks. Information services like World Wide Web (WWW) have also dramatically increased the usage of these networks. A growing demand in the telecommunications industry to support voice communication anywhere and at any time led to the evolution of cordless telephones, pagers, cellular systems and personal communication services (PCS). Advances in wireless technology further spurred this growth. A similar requirement for *uninterrupted* access to information by the computer user community has resulted in a synergistic convergence of wireless technology and data networks. Improvements in wireless communication methods and the rapid proliferation of small portable computers provide a greater scope for the development of wireless services for data communication.

In this computing environment users carry small portable computers like laptops, palmtops and personal digital assistants (PDAs). These are general purpose computing devices which due to their small size are convenient for working in more than one location. They are characterized by small memory and screen sizes and storage capacity and low power consumption. Currently e-mail and web access features provided by these devices are usable but very limited and hence these devices are presently less common than cellular telephones.

These devices do not normally have access to the rich communication capabilities of desktop computers. Nevertheless, it is interesting to envision an environment in which a student, for example, can take a small portable computer to a classroom or a library and still access a high speed fixed network on the campus. Wireless connections can provide temporary access to network services at conventions, seminars and conferences. A typical scenario might be a user who is usually connected to the network at his or her desk but occasionally would like to use his/her computer in a conference room in another part of the building or even at another site. Such a user is said to be *mobile* whenever there is such network migration (although he/she may not be actually moving while using his/her computer). The *mobile user* may connect to the network in various ways: through an Ethernet connection, through modem or telephone links, or through a wireless connection. These users can subscribe to information services such as stock quotes, news headlines, and weather reports that can be delivered directly to their mobile unit.

A wireless connection further enhances the user's ability to move since the user is not constrained by wire. A wireless connection need not necessarily imply that the user is mobile or vice versa. The user may be stationary at his/her desk but accessing the network through a wireless connection, but wireless technology also enables users to retain their network connection even when they are actually moving, provided they remain within the range of some wireless access point.

Seamless integration of mobile users and wireless connections with the existing fixed network infrastructures is very important. The incredible growth of the web was feasible because the infrastructure to support it was already in place; the same needs to happen for wireless applications to become more common. There are a number of research issues ranging from hardware, through networking and system software, to end-user applications. These can be broadly classified as follows [21, 32, 36]:

- **Communications management issues:** Communications management issues are related to bandwidth management and channel access in the context of wireless communications. The wireless medium is characterized by low bandwidth and high losses when compared to the wired media. These limitations pose major technical challenges in the design and deployment of wireless networks. Strategies for efficient bandwidth allocation among different users are highly desirable. Coverage is also a problem to be dealt with [28].

- **Location management issues:** Location management issues include location tracking and routing arising from the need to deal with the changing user location. Location management will become even more important as wireless network interfaces become widely available. Once the computing devices are unconstrained by cable, it is likely that frequent user movement across networks will become more common, both between sessions and within sessions.

- **Service issues:** Mobile users need to be able to access services at different locations in a transparent manner [19, 29]. The ability of users to roam, *i.e.* to move from one network to another without advance notice, leads to potential breaks in services. New service architectures are needed to provide effective services in the face of user mobility [14].

- **Other issues:** Security and reliability pose major challenges in mobile and wireless environments. Other challenges include the need for energy efficient and low-power design of the components [60]. With increased number of users and applications, scalability also becomes an important issue. New approaches are needed in designing such network architectures and, since all the various components of the network must be managed efficiently, network management is also a major issue [2]. It will be very difficult to implement the future multimedia applications in such an environment because of the very high bandwidth requirements of such applications. Providing quality of service guarantees can be very difficult when a wireless medium is involved.

1.1 Communication Protocols

Central to network (and internetwork) communications is the notion of *protocols*. Protocols specify the details of data communication including the message formats and the rules two or more computers should follow to exchange messages. Protocols

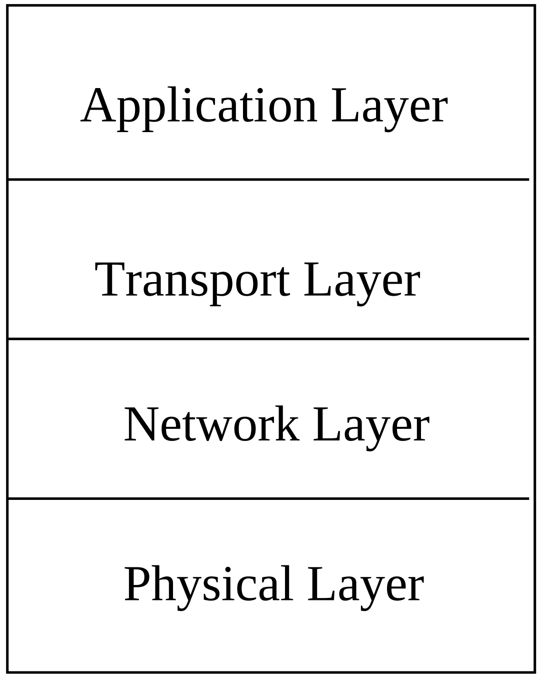

Figure 1.1: The TCP/IP Protocol Stack

also describe how a computer responds when a message arrives and how errors are handled. A *protocol stack* or *suite* is a collection of layered protocols. In two communicating machines, a layer on one machine may be considered to communicate with the corresponding layer on the other machine using the protocol of that layer. Each layer in a protocol suite defines a distinct service and implements a different protocol. A layer defines a level of abstraction in the protocol by grouping closely related functions. By decoupling layers, future changes made in one layer do not affect the design of the other layers.

Figure 1.1 shows the four layers of the TCP/IP protocol stack: the Physical layer, the Network layer, the Transport layer and the Application layer. This is the basis for much of the networking done today.

The *physical layer* comprises the communication channel through which the raw transmission of bits takes place. Twisted copper pair, coaxial cable, optical fiber, infrared beams or radio waves can all be used as transmission media. Different media differ in their propagation characteristics, such as error rates, bandwidths, etc.

Networks may be interconnected to form internetworks, and the functionality of interconnection in an internetwork is provided by the *network layer*. Internet Protocol (IP) is the network layer protocol in the TCP/IP protocol suite and it is responsible for routing datagrams across the Internet. IP provides only *best-effort* delivery service and does not guarantee reliable or ordered data delivery. IP leaves this functionality to the transport layer.

The *transport layer* provides a flow of data between two hosts for the application layer above [51]. The basic function of the transport layer is to accept data from the application layer, split it into packets, and then pass it to the network layer. These packets are reassembled at the other end and delivered to the application. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are the two major transport layer protocols in the TCP/IP protocol suite. TCP provides reliable data delivery to the application layer whereas data delivery via UDP is unreliable.

All applications reside at the *application layer.* FTP, TELNET, and WWW are some of the popular applications in the TCP/IP suite. These require reliable and ordered data delivery and use TCP to provide this functionality. Thus, there is no need to develop reliability mechanisms on a per-application basis. Network *sockets* are used by applications to access the underlying transport protocol. The socket interface facilitates reading and writing data streams.

1.2 Motivation for this Research

The current cordless telephones and cellular systems use wireless media for voice communication. Extending the use of wireless media for data communication offers the potential for a myriad of new services. Although wireless links offer an opportunity for users to access network services from different places at different times, the characteristics of the wireless links are significantly different from the current wireline links, making it difficult to design software that operates equally well in both environments. For example, wireless links have lower bandwidths and higher error rates than their wired counterparts. The TCP/IP protocol suite was developed based on certain assumptions that are no longer valid in this environment. Thus trying to accommodate wireless links into the Internet requires protocol enhancements at different layers of the TCP/IP protocol stack.

The focus of this research is TCP. Most of the applications on the Internet use TCP as the transport layer protocol for *reliable* data transfer across the network and so the performance of TCP directly affects application performance. For wireless connections to integrate seamlessly into the Internet, TCP should perform well in the wireless environment also. There have been recent proposals for transport layer support over wireless links but there is a need for a detailed performance study and a thorough evaluation of these proposals.

1.3 Goals and Objectives of this Research

The goal of this research was to investigate protocol issues that arise at the transport layer as wireless technologies are integrated into the existing network infrastructure. The investigation focused on TCP. Specific objectives of this research were:

- To study TCP's performance, specifically its reaction to loss in a real wireless testbed.
- To investigate the effect of various mechanisms within TCP on end-to-end connection performance.
- To evaluate possible solutions to problems encountered by TCP in a wireless environment, using simulations.

This research consists of two major phases. The first phase was a detailed measurement study of TCP's performance on a real wireless testbed. This helped to identify the problems encountered by TCP in a wireless environment. The measurement study also helped to understand the dynamics of TCP in a detailed manner. The second phase was consideration of a model called a "proxy" as a possible solution to problems encountered by TCP in a wireless environment. The performance of this proxy was evaluated using simulations.

1.4 Organization of the Thesis

Chapter 2 describes wireless networking in general, including the limitations of the wireless medium and how the wireless medium and mobility cause various problems at different layers of the protocol stack. Chapter 3 provides a brief review of TCP, followed by a description of existing literature on TCP in a wireless environment. Chapter 4 describes experiments conducted to evaluate the performance of TCP on a real wireless testbed. These results are explained by examining the behaviour of various TCP mechanisms in detail. In Chapter 5, the design of a model for a possible solution is outlined. Chapter 6 provides the detailed results of experiments performed using this model. Chapter 7 summarizes the work done and presents the conclusions.

Chapter 2

Data Networking with Wireless Technologies

Advances in wireless technology have made it possible to access data networks without a cable or any other wired media. Wireless links will not replace the existing wired links, but rather will complement them. The main advantage of wireless communication is that it facilitates tetherless access. Since installation costs are low, wireless communication provides an attractive alternative where cabling is difficult. A wireless connection provides new opportunities for mobile computing, which opens the door to a whole realm of applications. However, it is more than a simple migration of the existing wireless technology to support data communication. The nature of wireless technology brings in new challenges and new problems to be addressed. In this chapter, the various services offered by wireless media are briefly described, followed by a discussion of the limitations of wireless media and how these issues are being addressed at each layer in the TCP/IP protocol stack.

2.1 Wireless Services

Wireless services need to be deployed on a well-designed network. Such a network may consist entirely of wireless links, or wireless access can be provided to a conventional wired network. The generic term "wireless network" is often used to refer to a network which accommodates wireless links. Some different types of currently available wireless networks are described below:

- Paging networks have very low bandwidth and support one-way communication.

- Wireless LANs provide coverage for a local area (e.g., inside a building) and offer data rates from 2 to 10 Mbps [13]. The wireless host has a wireless interface card with an antenna that communicates with other hosts on the network through a wireless hub. A wireless LAN with only wireless links is possible, but the wireless hub normally is connected to a higher performance wired network. This gives rise to a heterogeneous environment, i.e. a network with high speed, low error-rate wired links and low speed, high error-rate wireless links. Examples of wireless LAN products include Lucent Technologies' WaveLan, Aironet's ARLAN, Motorola's Altair Plus-ISS, and Proxim's RangeLAN products.

- The current wireless wide area networks (WANs) are mobile radio networks providing services like e-mail. Although the area of coverage is large, the bandwidth available for data services is very small (19.2 Kbps). RAM Mobile Data, Ardis, and Metricom's Richochet network are examples of currently available wireless WANs.

- The existing cellular networks provide voice services to users with hand-held telephones. The Cellular Digital Packet Data (CDPD) protocol provides digital data transmission using the existing cellular infrastructure. The data packets are sent over signals already in use for voice communications by inserting them into the signal carrier in between the voice spurts. These networks offer low bandwidth (19.2 Kbps) and cannot handle a large number of users.

2.2 Wireless Access to the Internet

As described above, while most of the short range wireless systems seem to handle voice traffic, they need various enhancements to handle data traffic effectively [48, 55]. These systems can support only low data rate applications up to 19.2 Kbps [42] and also involve large message delays. The existing wireless networks satisfy

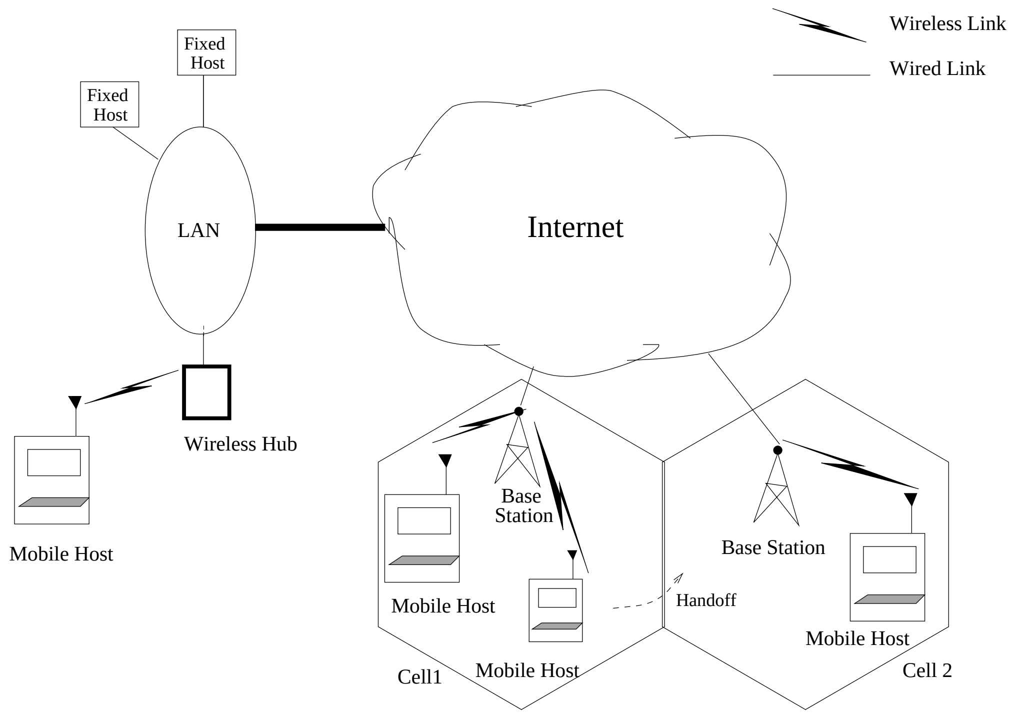

Figure 2.1: Network Model

two different types of user requirements [43]. On one hand, the requirement for relatively low-speed data services supporting mobile users over wide geographical areas is addressed by mobile data networks like RAM Mobile Data and Ardis. On the other hand, there are requirements for high-speed data services in local areas which are addressed by wireless LANs. There is an emerging need for high-speed data services at every location and every moment of time. These requirements are not satisfied by the types of networks mentioned above. This gives rise to a mixed environment.

It is likely that the architecture of future networks will evolve towards a high-speed wired backbone and various *mobile hosts* accessing this network through *base stations* as shown in Figure 2.1. A LAN in which the mobile hosts access the network through a *wireless hub* is also shown in Figure 2.1. Base stations and wireless hubs are attached to the wired network and provide connectivity to the wired network. These are usually equipped with a wireless transceiver and provide wireless access to users within their range. In such scenarios the wireless link will be the "last hop" or the "first mile" of the information superhighway. This research is concerned with such scenarios and does not consider networks with multiple wireless hops or a wireless backbone.

In such an environment, the coverage area for wireless communication is divided into *cells*, with a base station handling the communication with all the mobile hosts within a cell. The network model in Figure 2.1 shows two such cells and the associated base stations. As a mobile host moves from one cell to another (i.e. from the vicinity of one base station to the other), the task of transferring data between the wired network and the mobile host must be transferred to the new cell's base station. This process is called a *handoff* and introduces a delay in communication which is referred to as *handoff delay*.

A number of research projects have explored data networking in such a mixed environment. The PARCTAB [49] is a personal digital assistant (PDA) that communicates with a network through infrared (IR) transceivers. The infrared network is designed for in-building use, where each room becomes a communication cell. In contrast to the approach used by other PDAs, most PARCTAB applications run on remote hosts and therefore depend on reliable communication through the IR network. The infrastructure provides reliability as well as uninterrupted service when a PARCTAB moves from cell to cell. The applications are designed to accommodate the PARCTAB's low IR-communication bandwidth and small display area. The Bay Area Research Wireless Access Network (BARWAN) [40] project is developing a wireless overlay internetworking architecture along with a toolkit for mobile applications development. The *InfoPad* system [39] is built around a low-power, lightweight, wireless multimedia terminal that operates in an indoor environment. The goal of their project is to develop a platform and prototype for providing ubiquitous, wireless access to multimedia data. The MosquitoNet project [34] defines a "mobile people architecture" to put the person, rather than the devices that the person uses, at the endpoints of a communication session. Various tools such as the tracking agent and connectivity manager are being developed in this project to perform the routing functionality between people. The Monarch Project [20] addresses network support for wireless and mobile hosts. The goal of this project is to enable mobile hosts to communicate with each other and with wired hosts, transparently and adaptively, making the most efficient use of the best network connectivity available to the mobile host at any time.

While trying to provide services such as those described above, it should be noted that there are limitations for wireless communication. Some of these limitations are discussed in the following section.

2.3 Limitations of Wireless Communication

Wireless links are currently considerably slower than wireline links, and variations in signal quality make the communication less reliable. The major limitations of wireless communications can be attributed to the non-availability of spectrum and the highly error-prone wireless medium. Signal loss is related to propagation distance and physical surroundings. Interactions with typical building objects, including walls, metal, and even people, can affect how energy propagates, and thus the range and coverage a particular system achieves. Errors can arise from attenuation, co-channel interference or multipath fading.

Unlike physical media, such as copper or fibre, that provide a somewhat predictable and controlled transmission environment, the free space environment for wireless systems is constantly changing. Radio signals can be distorted by other nearby systems using the same frequencies. *Co-channel interference* refers to signal distortion due to signals from adjacent channels or other systems using the same frequencies.

When a signal reflects off or diffracts around objects, it takes multiple paths between the transmitter and the receiver. Because of reflections from buildings and other structures the signal arrives with different path lengths at the receiver. The arrival of such signals with different path lengths causes these signals to be out of phase with the direct signal, thereby deteriorating or even cancelling the signal. This effect is referred to as *multipath fading*.

The properties of the signals are also frequency dependent. At low frequencies, signals pass through obstacles, but the power falls off sharply with distance from the source. At high frequencies, the signals tend to travel in straight lines and bounce off obstacles. At about 8 GHz, the signals can also be absorbed by rain.

All these different types of effects ultimately result in a loss of data packets. These losses place a restriction on the maximum transmission rate. The interference effects are also of major concern when several networks operate simultaneously in the same place. Since the new wireless products have to co-exist with the already existing applications, they must deal with these interference effects. There are different wireless communication technologies to deal with these effects and these are described in the following section.

2.4 Wireless Communication Technologies

There are two major alternatives for wireless communication: infrared (IR) and radio frequency (RF). IR and RF are complementary and the choice of a particular method is clearly dependent on the requirements of the applications for which it is used. The technology used also significantly affects the propagation characteristics, bandwidth, power requirements and cost.

Infrared is the same technology used in household products such as remote controls for televisions and VCRs. The infrared signals have very high frequencies, of the order of 10^{12} Hz, which is just below visible light. IR signals cannot propagate through opaque objects and hence IR is inherently a "line of sight" technology. Since sunlight has a good portion of energy in the infrared region, the extreme intensity of solar radiation precludes the use of IR outdoors. Therefore IR is best suited for directional links and small interior rooms. The fact that IR does not propagate through opaque objects can also be used as an advantage. Infrared networks can be designed such that a system in one room of a building will not interfere with a similar system in adjacent rooms.

While IR is suitable for in-room, short-range wireless LANs, RF is suitable for longer range, high-capacity, full-coverage applications like building-wide wireless LANs. RF is also used for wide-area wireless networks like data on cellular telephone and packet radio networks. Since the radio frequency spectrum is a valuable and limited commodity, most countries regulate and license the use of radio frequencies. The frequencies 902-928 MHz, 2400-2483 MHz, 5725-5850 MHz that are part of the Industrial, Scientific and Medical (ISM) bands do not require licensing in North America. These are usually used for cordless telephones, garage door openers, wireless hi-fi speakers and security doors. The 915 MHz band is already crowded with other equipment and thereby prone to greater interference. The higher bands require more expensive electronics and are subject to interference from microwave ovens and radar installations. As mentioned earlier, the propagation characteristics vary by frequency band and increase in frequency implies greater multipath fading effects. Propagation above 3 GHz is mainly short range and line-of-sight (LOS). However, the higher frequencies offer higher bandwidth and consequently higher data rates (above 5 Mbps).

The 915 MHz and 5.8 GHz bands are available in the United States and Canada but are not available in most other countries. Systems designed for these frequencies can be used only in North America. The 2.4 GHz band however, is available almost world-wide [6]. Moving up in the frequency band means facing more challenging technology issues. From an end-user point of view, devices operating in the 2.4 GHz band will be more expensive than devices operating in the 915 MHz band, and devices operating in the 5.8 GHz band are more expensive than devices operating in the 2.4 GHz band. This results in a heavier utilization of the lower frequency bands, giving rise to greater problems of interference at lower frequencies.

Spread spectrum technology can be used in these ISM bands mentioned above to reduce interference and thereby isolate RF transmissions. A spread spectrum signal is created by modulating the signal with a spreading sequence (a code consisting of a series of binary pulses), or by hopping the frequency of the carrier signal [54]. There are two major types of spread spectrum techniques: frequency hopping spread spectrum (FHSS) and direct sequence spread spectrum (DSSS). More detail about these two types of spread spectrum techniques can be found in [30, 45].

Many spread spectrum systems can share the same frequency band, provided that each system uses a unique spreading code to reduce interference between the different systems. Only the receiver with the same spreading code as the transmitter can reassemble, or despread, the noise-like signal to recover the broadcasted information. Thus, spread spectrum systems can tolerate a level of interference higher than conventional systems.

2.4.1 Multiple Access Techniques

Multiple access techniques refer to the channel access techniques used for sharing a channel among multiple users. These are needed to provide more efficient usage of the network and to maximize network capacity. There are three major multiple access techniques: Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA) and Code Division Multiple Access (CDMA).

In FDMA the frequency spectrum is divided into a number of smaller channels. Each channel is assigned to a different user. Hence, the channels are the frequency bands. Since the total available bandwidth is divided into a number of smaller channels, FDMA is inefficient in spectrum utilization with bursty data traffic.

In TDMA each user has exclusive access to the entire channel periodically for a fraction of time, and thus each user's transmission is intermittent. In this case, the channels are the time-slots. With this scheme a remote terminal knows when it is allowed to transmit or receive and it can enter power saving states during the periods when the channel is used by other members of the network.

CDMA uses spread spectrum techniques to provide multiple access. Different users use different code sequences and each user can transmit over the entire available frequency range. CDMA is more robust in the presence of multipath and hence has the potential of allowing more users than either TDMA or FDMA [33]. Although CDMA is advantageous for bursty traffic, TDMA may be preferred when there are different user requirements in terms of service. The complexity and cost of the circuitry involved using CDMA should be considered against the net benefit anticipated in the system. The use of CDMA is not viable in a low-traffic-density area.

There is also a recent open standard under development called Vector Orthogonal Frequency Division Multiplexing (VOFDM) [57] for broadband wireless in the 25GHz range supporting data transmission rates up to 50Mbps.

The different frequency bands and multiple access techniques are not sufficient for accessing the Internet using a wireless link. They must co-exist with the TCP/IP protocol suite.

2.5 Protocol Support for the Mixed Environment

The use of wireless technologies requires more than just replacing the physical layer of the wired networks by the air interface. The packet loss due to the wireless links, along with the mobility of the user, must be dealt with while accessing the Internet using a wireless link. This requires protocol enhancements at all the layers of the TCP/IP protocol stack. There are a number of requirements of the protocols for the mixed environment described earlier. They must enable interoperation among different infrastructures (eg. wireless and wired), and deal with the unpredictability of user behaviour, network capability and computing platforms [2]. The protocols should be scalable with respect to heterogeneity, address space, bandwidth, geographical dimensions and number of users. The protocols should also deliver maximum independence between the network and the application from the users' viewpoint. It shouldn't be incumbent on the user to reconfigure or reboot each time the mode of communication access changes. Protocol enhancements at the different layers of the TCP/IP protocol stack are discussed in the following subsections.

2.5.1 The Physical Layer

Wireless differs significantly from wired transmission media in terms of error rate and channel capacity. Although wireless technology has been around in the telecommunications industry for voice services, new design decisions must be made for data transmission over the wireless channels. Although the usual channel access techniques like FDMA, TDMA, and CDMA can be applied to wireless data communications, problems such as signal fading and interference must be dealt with. Since bandwidth is a limited commodity, bandwidth reuse (how to partition the available bandwidth efficiently among individual users) is also a major issue in wireless communication.

In the TCP/IP protocol stack, the *link layer* is usually considered a part of the physical layer and is responsible for transmitting data reliably over the raw physical medium. This is accomplished by sending the data as sequential frames and processing the acknowledgment frames sent back by the receiver. The link layer consists of two sub-layers [53]: Media Access Control (MAC) and Logical Link Control (LLC). The MAC sub-layer is responsible for framing and checksumming. For multiple access media, MAC protocols also arbitrate access to the shared medium. The LLC is responsible for multiplexing different network layer protocols on the same link and providing recovery from lost frames. It is difficult to use Ethernet-like protocols such as Carrier Sense Multiple Access/Collision Detect (CSMA/CD) as a MAC protocol in the wireless medium if the transmitter simultaneously listens on the same channel on which it is transmitting. Since the transmitters are usually mobile, token-based protocols such as token ring cannot be used. Because of the presence of a very lossy medium and low bandwidth there is a need for the development of new link layer protocols [1]. Various link layer error recovery mechanisms are being proposed in the IEEE 802.11 standard. The standard will have a single MAC layer for the following physical layer technologies: Frequency Hopping Spread Spectrum, Direct Sequence Spread Spectrum, and Infrared.

2.5.2 The Network Layer

As mentioned earlier, the functionality of interconnection in an internetwork is provided by the network layer. In this mixed environment, some of the hosts in the network are mobile. The user may be mobile between sessions or within a session. In the first case, the mobile user moves to a different network and starts a new session. Here the problems are due to host relocation rather than the mode of access of the user. The use of wireless also facilitates mobility within sessions (for example, the user might move between cells while doing a file transfer or browsing the Web). This gives rise to problems due to both host relocation and the mode of access.

In both the above cases, the mobility of hosts introduces problems in the routing of packets. Since the functionality of routing of packets in an internetwork is provided by the network layer, location management should be addressed at the network layer [22, 14]. IETF Mobile IP is a solution proposed for supporting mobility in the Internet [44, 15]. Since the Mobile IP protocol does not depend on the characteristics of any particular subnetwork protocol, the same mechanism will work for both wired and wireless media.

2.5.3 The Transport Layer

While problems due to the communication medium are addressed at the physical layer, and problems due to mobility are addressed at the network layer, the transport layer should address both these problems. Transport layer protocols should be able to adapt to these new problems. They should be able to recover quickly and efficiently from losses on the wireless links and handoff delays. Existing transport layer protocols can operate unmodified in the presence of wireless links and host mobility, but at a cost of significant degradation in performance. The presence of a very lossy and variable bandwidth transmission medium invalidates some of the basic design assumptions made by protocols like TCP. Since this research is related to TCP, the various mechanisms in TCP and the problems encountered at the transport layer due to wireless media are very important. These are discussed in detail in Chapter 3.

2.5.4 The Application Layer

The lossy nature of the wireless medium may have a disrupting influence on application performance. Existing applications which depend on network support typically hang when network connectivity is disrupted [7]. In most cases, the user should not find any drastic difference between the wired network and the wireless link. Host mobility introduces additional problems due to the dynamic nature of system state. It is difficult to achieve full application transparency because of these problems.

Schillit *et al.* [49] suggest "dynamic customization" of applications, i.e. the system modifies application preferences in response to changes in the physical environment. For example, a web browser can automatically stop loading images from web pages when the bandwidth is low and resume loading images when the user moves to a high bandwidth environment. Application developers can exploit application-specific knowledge to address resource constraints but this should not result in modifications to software on the fixed hosts. It may be desirable to offer alternative actions to the user.

Mobisaic [56] is a World Wide Web information system designed to serve users in a mobile wireless computing environment. Mobisaic extends the web by allowing documents to both refer and react to potentially changing contextual information, such as current location in the wireless network. The idea is to support information browsing of the immediate surroundings through queries.

Wireless Access Protocol (WAP)[47] is a standard being developed for providing Internet communications and advanced telephony services on digital mobile phones, pagers, personal digital assistants and other wireless terminals. The WAP specification enables content creation and services across a wide range of wireless networking technologies and standards. WAP services are developed using the Wireless Markup Language (WML) that is similar to HTML.

2.6 Summary

This chapter describes issues related to wireless communication technologies and the network support required for supporting wireless applications. The various classes of applications place different demands on the network. These give rise to different system design and performance considerations. Certain network architectures are better suited to certain applications than others. The choice of a particular transmission technology or a multiple access technique depends on the particular application. The wireless and mobile computing environment is likely to evolve into a more comprehensive network environment, with a very high speed backbone and many mobile users accessing the network through base stations over a wireless medium with frequent movement between networks. Mobile and/or wireless users want to run all the applications running on the fixed network, including data intensive applications such as FTP or Web browsing. This necessitates the support of mobility and wireless links in the Internet, which requires protocol enhancements at different layers of the protocol stack.

Chapter 3

TCP: A Brief Overview

If the underlying network is unreliable, error detection and recovery mechanisms are indispensable, but it is cumbersome to build such mechanisms within every application. The Transmission Control Protocol (TCP) provides a general solution by providing an interface for reliable data transfer, and thereby isolates the application programmer from the details of the underlying network. For this reason, TCP is a very important part of the TCP/IP protocol stack. Since the 1981 release of its specification [46], implementations of TCP have been augmented with several performance-enhancing mechanisms, such as flow control, congestion control, fast retransmission, and fast recovery [9, 23, 24, 8]. These mechanisms and their interactions have both improved the performance of TCP and increased its complexity.

The common implementation of TCP/IP has been distributed with the 4.x BSD (Berkeley Software Distribution) Unix releases and with the "BSD Networking Releases". This source code has been the starting point for many other implementations. Figure 3.1 (adapted from [51]) shows the chronology of the various BSD releases, indicating additions of important features with each release. The releases shown on the left side in the figure are publicly available source code releases, which contain all the networking code: the protocols, the kernel routines for networking interface, and many of the applications and utilities. The "Tahoe" TCP implementation incorporated performance enhancing mechanisms called *slow start, congestion avoidance* and *fast retransmit.* The "Reno" TCP implementation modified the sender to include the *fast recovery* algorithm. These algorithms are discussed in

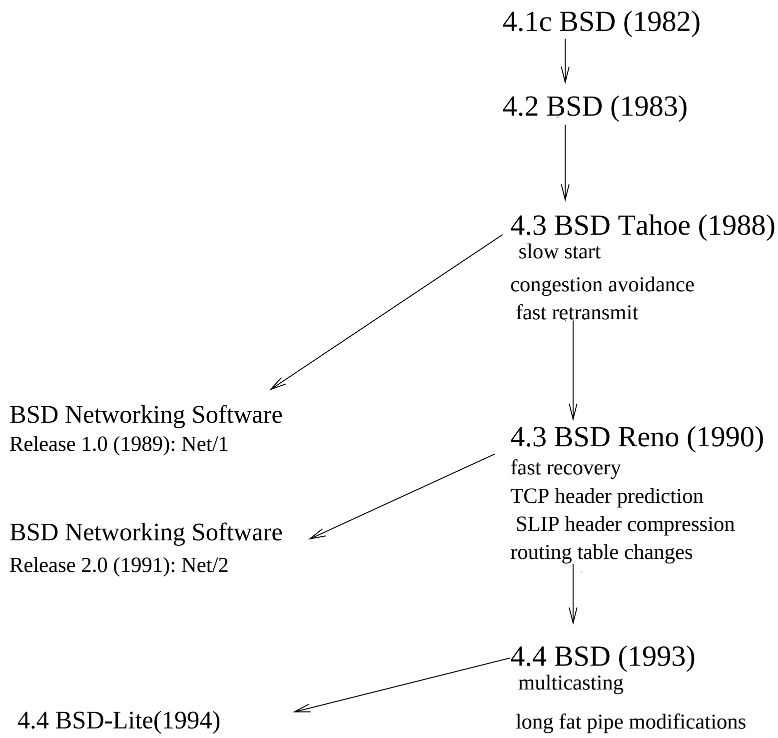

Figure 3.1: A History of BSD Unix Releases and Important TCP Features Added with each Release.

detail in later sections of this chapter.

As the dominant protocol on the Internet, TCP provides a *de facto* standard for host-to-host communication. In 1991 it was reported [11] that TCP packets make up roughly 80% of all wide-area network traffic, and that 6 out of 35 applications identified account for more than 96% of the bytes transmitted. These 6 applications were FTP, SMTP, NNTP, VMNET, TELNET and RLOGIN, suggesting that most of the traffic is caused by well-known TCP services. Hence, retaining TCP in the Internet is important. Although the traffic pattern in future networks supporting wireless access and mobile computing is expected to be different from the current fixed networks, remote information access and connectivity to the Internet will continue to be in great demand. In this chapter, after a brief review of TCP, problems arising at the transport layer from the introduction of wireless links into the Internet are discussed. This is followed by a discussion of the different approaches to enhance TCP performance in the presence of wireless links and mobility.

3.1 TCP Services

TCP operates on top of the unreliable IP layer and provides a connection-oriented reliable byte-stream service. The data bytes in the stream are numbered sequentially so that each byte has its own 32-bit *sequence number*. TCP uses the *connection*, which can be described by the four-tuple ⟨source IP address, source port number, destination IP address, destination port number⟩, as the fundamental abstraction. The source and destination *port numbers* identify the sending and receiving applications. TCP connections are full duplex and the TCP modules at each end of the connection communicate by sending TCP *segments*. A TCP segment consists of a header, which is usually 20 bytes (without *options*), and data, which can be of variable size. The sequence number field in the packet header contains the sequence number of the first data byte carried by the segment. TCP maintains a *checksum* on its header and data to detect any modification of data in transit. TCP segments are transmitted as IP datagrams over the network. Since the IP layer does not guarantee reliable delivery, the TCP segments can arrive out-of-order at the receiver or be duplicated.

Reliable delivery implies that all the data packets are delivered to the application program in the correct order. This requires mechanisms to recover from lost packets, to regulate packet flow, and to control congestion in the network. TCP provides reliable communication by using checksums, sequence numbers, acknowledgements (ACKs), sliding windows, retransmissions, and mechanisms for flow control and congestion control. These are described in the following sections.

3.2 Acknowledgement Scheme

To ensure that all packets arrive at their destination TCP requires that acknowledgements (ACKs) be sent back to the sender. The *acknowledgement number* indicates the next sequence number that the sender of the acknowledgement expects to receive [51]. To improve performance, a *cumulative* acknowledgement scheme is

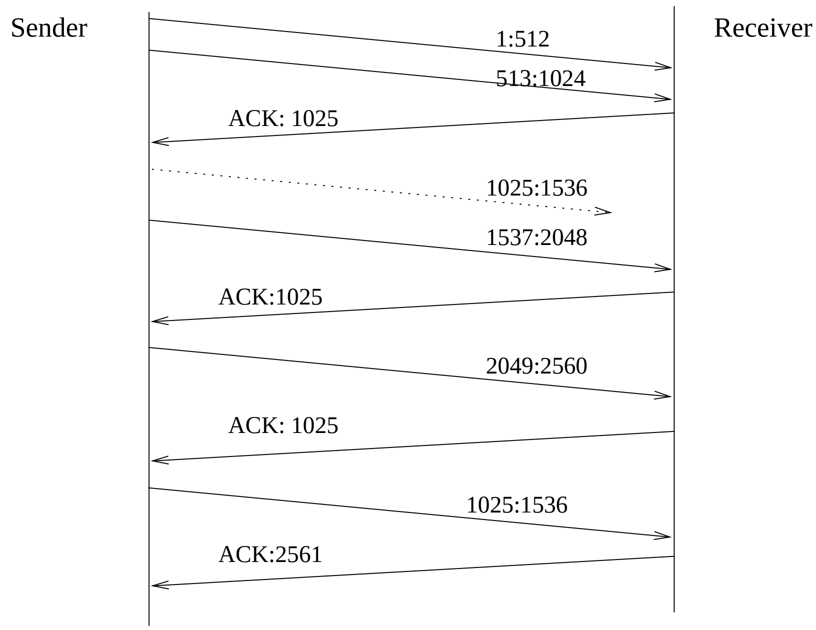

Figure 3.2: The TCP Acknowledgement Scheme

employed. This means that an acknowledgement indicates to the sender that all the bytes up to but not including that byte have been successfully received. If the receiver sends an ACK for sequence number 513, for example, this means that all the bytes up to sequence number 512 have been received.

To illustrate how cumulative acknowledgement works, assume that the sender is sending segments of 512 bytes as shown in Figure 3.2. The first segment covers sequence numbers 1-512, the second covers the range 513-1024, and so on. Assume that the third segment is lost in transit. When the receiver receives the first two segments it generates an ACK for the second segment. Since the third segment (1025-1536) is lost, the arrival of the fourth segment (1537-2048) causes the receiver to generate a duplicate ACK. This once again acknowledges the sequence number 1024, the highest *in-order* sequence number received so far. The fourth segment is placed on the *reassembly queue* at the receiver. Until the third segment is received, each additional segment received is cached at the reassembly queue and a duplicate ACK is generated. Once the third segment arrives, TCP uses that segment and the ones in the reassembly queue to reassemble the longest in-order sequence of segments possible and generates an ACK for the highest sequence number in that sequence (if the third segment is received after the fifth, an ACK is generated directly for the fifth segment).

TCP does not send an ACK the instant it receives a segment. Instead, it delays the ACK, so that if there is data going in the same direction as the ACK, the ACK can be sent along with, or "piggybacked" with, the data. Thus the ACKs are *delayed*. To prevent the receiver from waiting forever, the ACKs are delayed for 200 msec after which they are generated regardless of whether or not there is data going in that direction. But when an out-of-order segment arrives, a duplicate ACK is generated immediately.

3.3 Retransmissions and Timeouts

TCP uses packet retransmission as the final element in ensuring reliable delivery. If an ACK for a segment is not received within a given timeout interval, the segment is retransmitted. Because of the variability of the networks that make up an internetwork system, this *retransmission timeout* value is calculated dynamically. The retransmit timeout (RTO) value is updated constantly, based on the measured round trip time (RTT) and its variance. The details of the calculation of the RTT and RTO values are given in [27, 23, 50].

3.4 Flow Control

In an internetwork environment, devices of various sizes and speeds communicate with each other. For example, a minicomputer may be accepting data from a supercomputer. The sender is faster than the receiver and so packet overflow may occur at the receiver, which would result in packets being dropped at the receiver. To prevent overflow, flow control mechanisms are used to regulate the transfer of data [16].

TCP is called a window-based protocol because it uses windows to control the number of bytes that can be in transit. Flow control in TCP is based on a sliding window mechanism. The window size defines the maximum amount of unacknowledged data that the sender can have outstanding. If the window size is W and the last acknowledged byte is A, the sender can send bytes A+1 to A+W but must then stop sending until it receives an ACK from the receiver. As segments are acknowledged, this window slides along the stream, transmitting new segments.

The receiver specifies the number of additional bytes it is ready to accept through a *window advertisement*. This is an indication of the receiver's current buffer size. When the window advertisement is increased, the sender increases the size of the sliding window, and when the window advertisement is decreased, the sender decreases the size of the sliding window. This allows the receiver to restrict transmission until it has sufficient buffer space to accept more data and thus prevents a fast host from sending more data than can be read by the receiving host.

3.5 Congestion Control

Congestion in a network results in degraded performance. Congestion is perceived by the endpoints as an increased delay. Congestion control mechanisms work towards reducing or preventing this deterioration in performance. Normally, the endpoints do not know where congestion has occurred or why.

TCP takes steps to control congestion in the networks through algorithms called *slow start* and *congestion avoidance* [23]. The basic idea is to control the rate of sending packets to the network: to reduce the rate when congestion is detected and increase the rate exponentially as allowed. At any point in time, the TCP sender is in either the slow start phase or the congestion avoidance phase. The rate of data transmission increases more aggressively in the slow start phase than in the congestion avoidance phase. A TCP sender is in the slow start phase whenever the sender first starts transmitting data or immediately after a retransmit timeout. The sender shifts from the slow start phase to the congestion avoidance phase at a certain threshold or immediately after a *fast retransmit* (fast retransmit is discussed in detail in Section 3.6). This mechanism adds another window to the sender, called the congestion window *cwnd*, and a slow start threshold size, *ssthresh* [23].

The ssthresh value dictates when the sender shifts from the slow start phase to the congestion avoidance phase. The congestion control algorithm is described very briefly in the code snippet below:

```
if (cwnd <= ssthresh)
    slowstart phase:
         cwnd increases exponentially
else
    congestion avoidance phase:
         cwnd increases linearly
```

The amount of data the sender is allowed to transmit, referred to as *win*, is limited by the minimum of the congestion window and the advertised window of the receiver (*snd_wnd*).

$$win = min(cwnd, snd\_wnd); \tag{3.1}$$

Cwnd is initialized to one segment when a connection is established. During the slow start phase, *cwnd* increases exponentially. The sender starts by transmitting one segment, and when the ACK is received, increases the *cwnd* to two and two segments can be sent. When each of these two segments is acknowledged, the congestion window is increased to four segments. This exponential increase continues until *cwnd* becomes greater than or equal to *ssthresh*. At this point, the sender enters the congestion avoidance phase, during which *cwnd* increases linearly.

Congestion is assumed upon a retransmission timeout or the receipt of three duplicate ACKs. When congestion is detected by a retransmission timeout, the *ssthresh* value is adjusted to one-half of the current window (the minimum of *sndwnd* and *cwnd*). *Cwnd* is reinitialized to one segment and the slow-start phase is entered. In other words, the current threshold is too high and the sender is transmitting packets too aggressively, resulting in congestion which manifests as packet loss. This threshold is lowered so that the sender shifts from exponential increase to linear increase of its congestion window earlier to avoid congestion in the future.

3.6 Fast Retransmits and Fast Recovery

A *fast retransmit* mechanism is employed to facilitate earlier detection of a missing segment. Instead of waiting for a long retransmission timeout, upon receiving three[1] duplicate ACKs the sender infers that a segment has been lost [8, 51]. TCP retransmits, starting with the sequence number immediately after the sequence number specified by the duplicate ACK. Under the *fast recovery mechanism*, the sender enters the congestion-avoidance phase instead of the slow-start phase after a fast retransmit.

3.7 TCP in a Wireless Environment

The congestion control algorithm of TCP assumes that "on most network paths, loss due to damage is rare ($\ll 1\%$) so it is probable that a packet loss is due to congestion in the network" [23]. In the case of wireless links, this assumption fails to hold because there are many packet losses unrelated to congestion. Packets may be lost because of transmission errors caused by the lossy nature of the wireless link. Packets may also be lost over the wireless link when a mobile host moves out of reach of other cells.

When the wireless link causes a TCP connection to break down, few applications are sophisticated enough to continue the interrupted activity [31]. For example, a file transfer may have to be restarted from the beginning.

TCP's actions are effective in a wired environment where loss of a packet is rare, but when this is not the case (as in a network with wireless links) these actions need to be reconsidered. The implications of losses due to wireless links on the performance of a TCP connection are studied in this thesis.

[1] The sender does not know whether a duplicate ACK is due to a lost segment or due to reordering of segments, so the sender waits until a small number of duplicate ACKs are received. It is assumed that if three or more duplicate ACKs are received in a row, it is a strong indication of packet loss. If there is just a reordering of segments, there will be one or two duplicate ACKs before the reordered segment is processed. So the above assumption is reasonable.

3.8 Related Work

As mentioned in the previous chapter, problems due to the wireless link and the mobility of the users must be addressed at the transport layer. Work related to this research is of two types: studies of TCP performance in wireless and mobile environments, and proposals for modifications to TCP in such environments.

The work of De Simone *et al.* [17] can be classified as the first type. They investigated the effect of link-level retransmission on transport layer performance using simulation and analytical techniques. The wireless link implements a retransmission protocol at the link layer. They showed that the interaction between the link layer and transport layer retransmissions can lead to degraded TCP performance, especially as the error rates become more significant. For this reason the link layer should have knowledge of the timeout values and retransmission policies of the transport layer.

Caceres *et al.* [12] studied the effects of mobility on TCP through measurement in a wireless networking testbed. A TCP connection was initiated between a mobile host and a fixed host. The packets from the mobile host were routed to the fixed host via a base station which was connected to the same wired network as the fixed host. Motion across cell boundaries was simulated in software. They found that the throughput dropped significantly with motion because TCP assumes that the delay resulting from handoff is due to congestion and thus reacts inappropriately. They also investigated the effect of having a fast retransmit mechanism. Their measurements indicated that delay in communication during a handoff was reduced from 650 msec to 50 msec when the fast retransmit mechanism was used.

Manzoni *et al.* [35] used a simulation model which included key features of TCP and Mobile IP to predict the impact on transport layer performance of round-trip delay, handoff delay and window size. They suggested that some interaction between the transport layer and the network layer is needed to alleviate the impact of mobility on the transport layer throughput. In particular, they issued an explicit notification from the network to the transport layer to extend the TCP slow-start phase during handoffs.

The above studies do not address the issue of packet loss resulting from wireless links. Fast retransmit solves only a part of the problem and may not be effective in the case of wireless links. As mentioned in the previous section, fast retransmit uses the arrival of three duplicate ACKs as an indication of packet loss. In the presence of lossy wireless links, the ACKs themselves may be lost which makes the fast retransmit mechanism ineffective. A timeout still occurs in this case.

3.9 Proxy: A Possible Solution?

The main idea behind using a proxy is to address the problems as a result of the wireless link locally. The proxy splits the TCP connection into a wired part and a wireless part, and performs local retransmissions on the wireless link. Even with the proxy handling the retransmissions on the wireless link, the sender need not be aware of the presence of a proxy. The base station, through which the wireless host accesses the fixed network, can act as the proxy, and the host on the fixed network need have no knowledge of the proxy. In other words, the use of a proxy is transparent to the sender and the receiver. This approach makes an implicit assumption that the lossy wireless link is at or close to the last hop between the sender and the receiver. Since modification of software on existing fixed hosts is not desirable, it is a good idea to have a proxy handle the losses on the wireless link.

In the following paragraphs, some of the approaches which use proxies are discussed.

One such approach, referred to as Indirect TCP (I-TCP) [3], allows a mobile host to communicate with the fixed network over a transport layer connection through its current mobile support router. The functionality of the mobile support router is similar to a base station base station. As shown in Figure 3.3, the TCP connection is split into two separate TCP connections: one between the fixed host and the base station over the wired link, and the other between the base station and the mobile host over the wireless link. In other words, the TCP connection with the mobile host

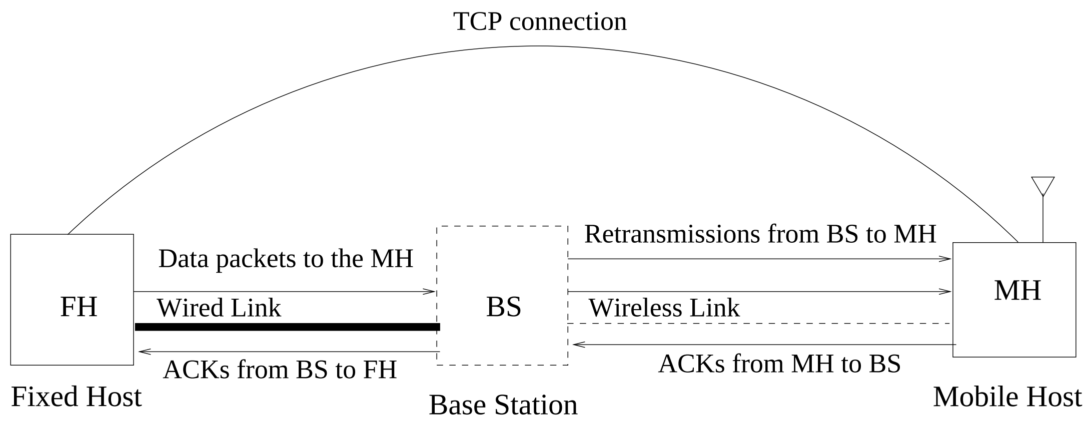

Figure 3.3: The I-TCP Approach

is actually established by the base station on behalf of the fixed host. If the mobile host moves to another cell during the lifetime of the TCP connection, the new base station takes over the connection from the old base station (i.e. the connection is not broken by a handoff). Thus the fixed host sees only an image of the mobile host that resides on the base station. When the mobile host moves to another cell, this image is handed over to the new base station. Since the connection endpoints for both the wireless and fixed parts of the I-TCP connection do not change after a move, there is no need to re-establish the connection at the new base station. The base station fakes an image of the mobile host for the wired part of the connection by using the IP address and port number of the mobile host. Mobile IP takes care of the routing of packets to the new base station when the mobile host switches cells. When the mobile host moves from one cell to another, the state of all its active I-TCP connections is handed over to the new base station which effectively restarts the connection in the new cell without any participation from the communicating end points.

Although it works, there are several disadvantages to the I-TCP approach. The semantics differ from TCP in the sense that the acknowledgements are not end-to-end and the sender receives an ACK from the base station before the data reaches the intended receiver. Also, mobile host applications must use special I-TCP calls instead of the regular socket system calls, meaning the applications must be recompiled using the I-TCP libraries. It also appears that considerable processing is being done during a handoff. Every packet must go through the protocol stack four times: once at the sender, twice at the base station, and once at the receiver.

Yavatkar and Bhagwat [59] introduced a new session layer protocol on top of TCP at both the base station and the mobile host. The session layer protocol is designed to exploit the available knowledge about both wireless link characteristics and host migration. They considered two alternatives: the Mobile Host Protocol (MHP) and the Selective Repeat Protocol (SRP). The Mobile Host Protocol (MHP) establishes two TCP connections: one from the mobile host to the base station and the other from the base station to the fixed host (this is similar to I-TCP). The Selective Repeat Protocol (SRP) uses a specialized transport protocol for the wireless link. The protocol over the wireless hop uses its own flow and error control mechanisms designed and optimized specifically to address the lossy and erratic delay characteristics of the wireless link. SRP uses a selective ACK when an out-of-sequence segment is received. This reduces the number of segments being retransmitted on the wireless link. Their testbed consisted of the Internet and a simulated wireless link. They also simulated the handoff as a delay during which the IP software drops all the packets.

Balakrishnan *et al.* [5] suggested caching packets at the base station and performing local retransmissions of TCP packets across the wireless link as shown in Figure 3.4. They modified the routing code at the base station by adding a "snoop" module which maintains a cache of TCP packets sent from the fixed host that haven't yet been acknowledged by the mobile host. The snoop module also keeps track of all the acknowledgements from the mobile host. When a packet loss is detected, the lost packet is retransmitted from the cache at the base station, thus hiding the packet loss from the fixed host by not propagating duplicate acknowledgements. This prevents unnecessary congestion control mechanism invocations. They used

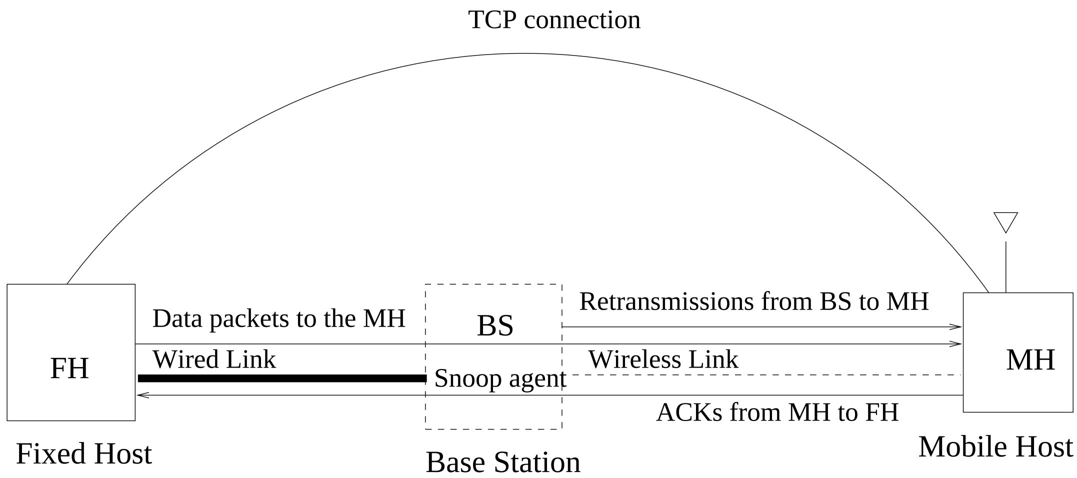

Figure 3.4: The Snoop Protocol

the selective acknowledgement (SACK) scheme for data transfer from a mobile host to a fixed host. This requires a minor modification to TCP at the mobile host to handle the SACK. They also presented a comparative experimental analysis of the snoop protocol and other schemes for improving end-to-end TCP performance and showed that the snoop protocol does improve end-to-end TCP throughput [4].

This scheme has several significant advantages over the I-TCP approach. It does not sacrifice end-to-end TCP acknowledgement semantics, as does I-TCP. There is effectively one TCP connection, whereas there are two TCP connections in the I-TCP approach. The snoop module is at the network layer, in contrast to the I-TCP software which is at the transport layer. Handoffs are based on multicast, and observed handoff times are in the range of 10 to 25 msec, as compared to 265 to 1400 msec in the I-TCP approach. It is to be noted that the amount of state information stored for each connection at the base station has an impact on the scalability of these approaches.

Although the general idea of a proxy is a viable solution, there are problems with the currently available approaches as mentioned above. It is not clear to what extent the proxy mechanism contributes to improvements in performance. It is also necessary to come up with specific design parameters for a generalized proxy approach. This motivates a thorough performance study of the proxy approach.

3.10 Summary

Since the wireless link is usually the "last hop" or the "first mile" of the network, a reasonable design approach is to address the resulting problems locally. For example, since the packet loss is much greater on the wireless link, when compared to the wired part, by trying to solve this problem locally, unnecessary retransmissions all along the fixed network can be avoided.

The need for other nodes in the network to know about the lossy links should be minimized, if not eliminated. The amount of state information stored in the base station should be minimized for scalability reasons. Major modifications of software on existing fixed hosts is not desirable. For these reasons, the use of proxies in which modifications are required only at the mobile host and the base station (or last-hop router) seems to be a more appropriate solution. A separate protocol can be used at either the transport layer or the link layer which is specialized for the wireless link and is transparent to the fixed hosts.

Chapter 4

Measurement Study

The research carried out for this thesis consisted of two major phases. The first phase was a measurement study of TCP performance on a real wireless testbed. The second phase consisted of evaluating the impact of using a proxy through simulations. The measurement study is described in this chapter.

4.1 Measurement Study

4.1.1 Goals of the Measurement Study

The performance of TCP in a wired environment has been studied in detail by many researchers [61, 38, 23, 10]. The goal of this measurement study was to look at how TCP, a protocol designed for the wired environment, behaves in the presence of wireless links, and to interpret this behaviour in terms of the various mechanisms discussed in the earlier chapters. The study also sought to investigate the impact of TCP's actions on the end-to-end throughput of the connection.

4.1.2 The Experimental Testbed

The experimental testbed consisted of two hosts communicating with each other as shown in Figure 4.1. Host A (the sender) was a DEC Alpha 3000 workstation running the DEC OSF/1 operating system connected to a 10 Mb/sec Ethernet. Host B (the receiver) was an Apple PowerBook 520C laptop running the MacOS operating

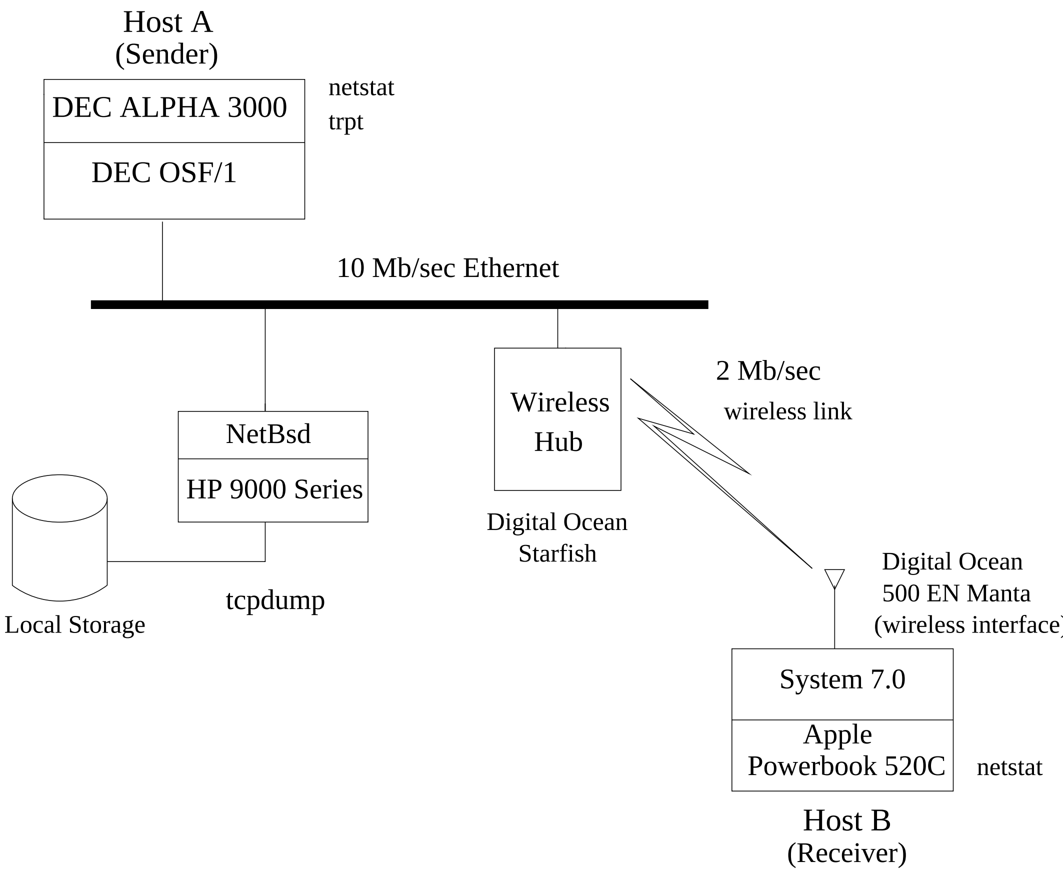

Figure 4.1: The Experimental Testbed

system, (version 7.0 with an emulated version of Mach2.5 Unix) accessing the Ethernet through a wireless interface (a Digital Ocean 500EN Manta wireless interface which communicates with a Digital Ocean Starfish wireless hub). The wireless link was rated at a raw speed of 2 Mb/sec and used spread spectrum communication operating in the 902-928 MHz range. Communication on the wireless link followed the IEEE 802.11 standard.

4.1.3 Measurement Tools

A TCP connection was established between Hosts A and B, across which data was sent. The network monitoring tool *tcpdump* [37] was used to collect traces of the

TCP packets flowing between the two hosts. To minimize interference, tcpdump ran on a separate machine attached to the Ethernet as shown in Figure 4.1 and saw only the packets on the wired segment of the connection. The traces were stored locally and analysed offline. Tcpdump records the headers of each packet, which provides information such as sequence numbers, acknowledgements, and window advertisements. A program called *netstat* was run at both the sender and the receiver, before and after each session of the experiment, and provided information such as number and type of packets sent and received. Some parameters of interest, such as the congestion window and timeout values at the sender, reside in the operating system kernel (specifically, in the TCP Protocol Control Block) and so cannot be accessed directly. To get this information the socket debug (SODEBUG) option was set in the application program to dump this information into a buffer within the kernel (*/dev/kmem*) and a program called *trpt* was used to read this information for subsequent analysis.

4.1.4 Performance Metrics

The following metrics form the basis of this study:

- **Completion time** is the time elapsed from the setting up of the TCP connection to its tearing down.

- End-to-end **throughput** is the data transfer rate over the lifetime of the connection, and is computed by dividing the total number of data bits transmitted by the completion time. The throughput computed is "effective throughput", in the sense that it does not take into account the extra data transferred as a result of retransmissions.

- The number of **retransmissions** is determined in the TCP packet trace by counting the number of re-occurring sequence numbers. Since the Ethernet segment is assumed to be loss-free, it is assumed that all retransmissions are due to packet loss on the wireless link.

- **Retransmission timeout (RTO)** values give an indication of the behaviour of TCP's retransmission mechanism.

- The number of **zero window advertisements** is determined from the trace of TCP packets. Zero window advertisements indicate that the receiver is unable to keep up with the data flowing from the sender, and inform the sender to stop transmitting data. The sender waits until the receiver sends a **zero window update**. This indicates to the sender that the receiver is willing to accept more data. Since zero window updates could fail to arrive, the sender will probe the receiver by using **zero window probes**. The number of zero window probes is obtained through *netstat*.

- The values of **cwnd** and **ssthresh** during the lifetime of the connection show the behaviour of the congestion control mechanism. These values are recorded at the sender by using the SODEBUG option in the application program and are read from the kernel buffer using *trpt*.

4.1.5 Experiments

From some initial experimentation it was determined that the error rate on the wireless link is a function of the distance between the mobile host with the wireless interface and the wireless hub. For these experiments, different error rates were obtained by placing the mobile host at different distances from the wireless hub. The distances are categorized into three ranges: close, medium and far [26]. The distance referred to as "close" was achieved by placing the mobile host on the same bench, at a distance of roughly 0.25 m from the wireless hub; the distance referred to as "medium" was achieved by taking the mobile host roughly 15 m from the wireless hub; and the distance referred to as "far" was achieved by taking the mobile host roughly 25 m from the wireless hub. As the data shows, these three ranges correspond to different error rates. Experiments were also performed with a distance greater than 30 m (referred to as "very far"), although at this distance the TCP connection has so many problems that it does not terminate, even after 45 minutes.

This limits the results that can be obtained from this case. The presence of additional obstacles between the receiver and the hub at long distances (walls, doors, people, etc.) also has an effect on error rates. Experiments were conducted during evening hours of light load, so that the effect of other network traffic on the wired portion is minimal. For each experiment a TCP connection was set up between Host A (the *sender*) and Host B (the *receiver*) and a 10 Mbyte data file was transferred using an application program which writes to the TCP streams socket. FTP was not used because it uses two TCP connections and also brings in its own control packets and other protocol overhead. To reduce the effect of the uncontrolled factors, five different runs were conducted in each of the ranges, and the values reported are the averages across the five runs.

4.1.6 Results

Figure 4.2 shows the sequence numbers of the packets flowing on the network, as seen in the traces. This figure corresponds to that portion of the trace between the 360th and 420th packets as recorded by tcpdump. The dip in the curve indicates that packets have sequence numbers that have already been seen on the network. This means that these particular packets are retransmissions.

The effect of increasing the rate of errors (by increasing distance) is shown in Table 4.1. As expected, the loss of packets causes retransmissions to occur and this results in increased completion time and decreased throughput. These effects are entirely due to the loss of packets on the wireless link, since the level of network utilization is low. At the close range, the entire 10 Mbytes is transferred in about 348 seconds. In the medium range, the additional 27 retransmissions add about 8192 bytes. Assuming the rate at which the sender pumps data into the network is similar to the previous case, this should account for at most an additional 1 sec. The difference in the completion time as shown in Table 4.1 is, however, much greater. The increase in completion time observed is thus due not only to the increase in number of retransmissions but also to the retransmission mechanism.

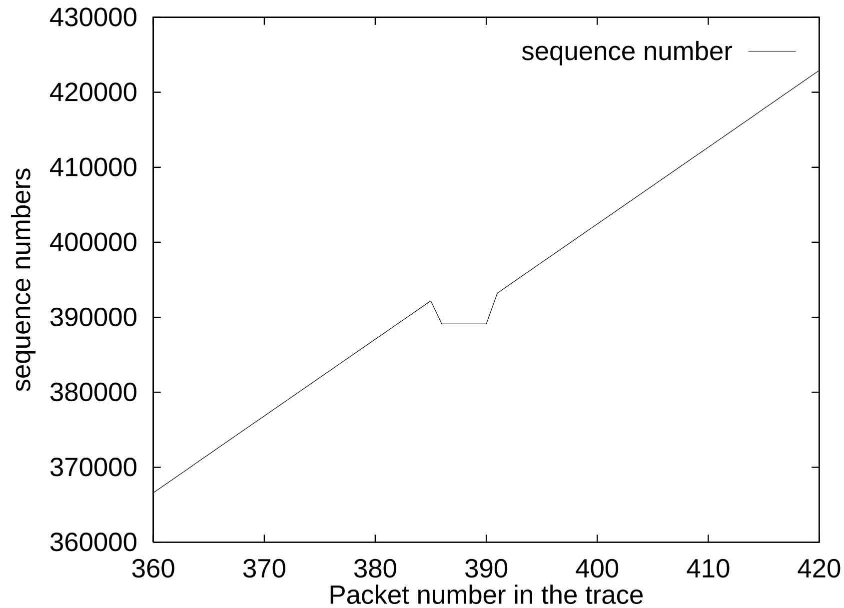

Figure 4.2: A Partial Trace of the Packets

Table 4.1: Results From the Regular Load

Range	Retransmissions	Completion Time (seconds)		Throughput (Mbits/sec)	
	Average	Average	Std.dev	Average	Std.dev
Close (approx. 0.25m)	4.0	347.67	9.49	0.228	0.007
Medium (approx. 15m)	31.6	616.83	42.08	0.130	0.008
Far (approx. 25m)	71.0	1063.78	40.89	0.078	0.002

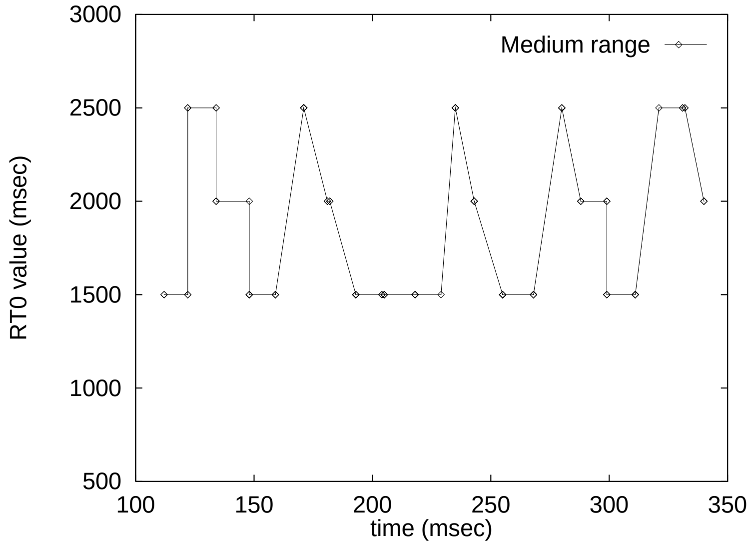

Figure 4.3: Behaviour of the Retransmission Timeout Variable (Medium Range)

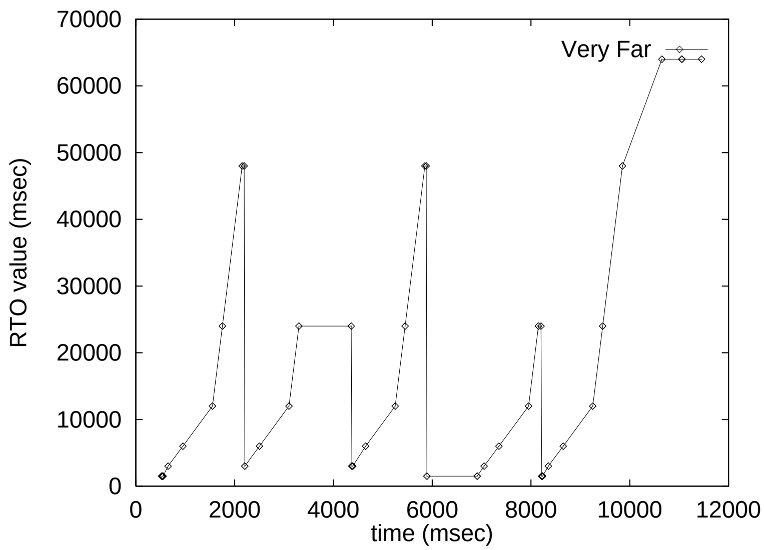

Figure 4.4: Behaviour of the Retransmission Timeout Variable (Very Far, approx. 30m)

This prompted an investigation of the internals of the retransmission mechanism and the congestion control mechanism. To illustrate this behaviour better, the behaviour of the retransmission timeout (RTO) values at the sender are compared for the medium range and the very far range. Figures 4.3 and 4.4 show the last 100 RTO values for the medium range case and for the very far case, respectively.

It is seen from these figures that the RTO value reaches a maximum of 2.5 seconds

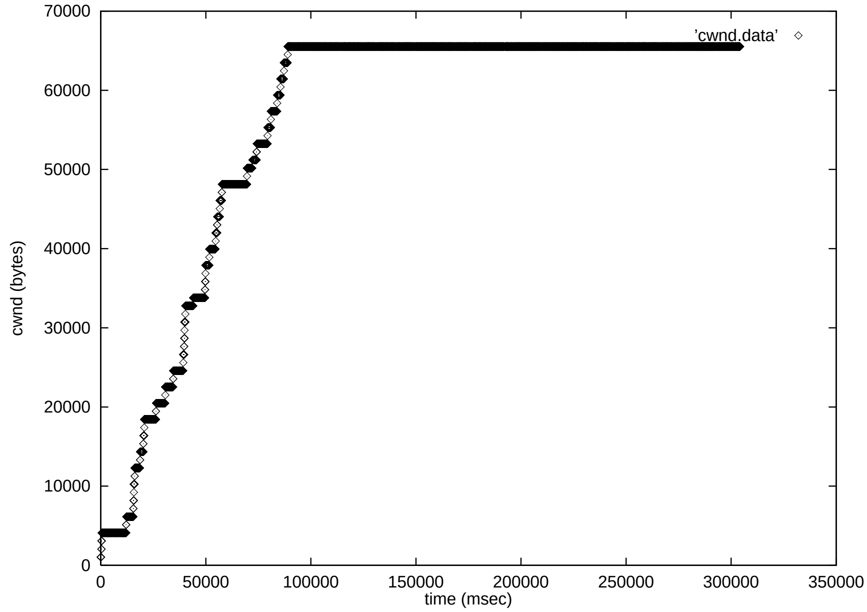

Figure 4.5: Behaviour of the Congestion Window (Close Range)

for the medium range and a maximum of 64 seconds for the very far range. This means that for the very far case the sender must wait for 64 seconds (maximum RTO value) before retransmitting the lost packet.

From Figure 4.4 it can also be seen that the RTO value doubles whenever there is a lost packet. This is appropriate behaviour from the sender's point of view, since the sender assumes that the packet loss results from congestion. So the sender waits a longer time to retransmit the lost packet instead of exacerbating the congestion with retransmissions. Since this loss is due to the wireless link, however, this is an inappropriate response because the sender is forced to wait a longer time to retransmit the lost packet instead of trying to recover from the loss more immediately. In other words, rather than backing off (as would be appropriate for congestion) the sender should retransmit the lost packet immediately.

Plots of the congestion window (cwnd) value throughout the duration of a connection for close and medium ranges are shown in Figures 4.5 and 4.6 respectively. In Figure 4.5, cwnd increases until it reaches a maximum value, whereas in Figure 4.6 cwnd is reinitialized to one segment due to perceived congestion. So the sender reduces the rate of data transmission which results in decreased throughput.

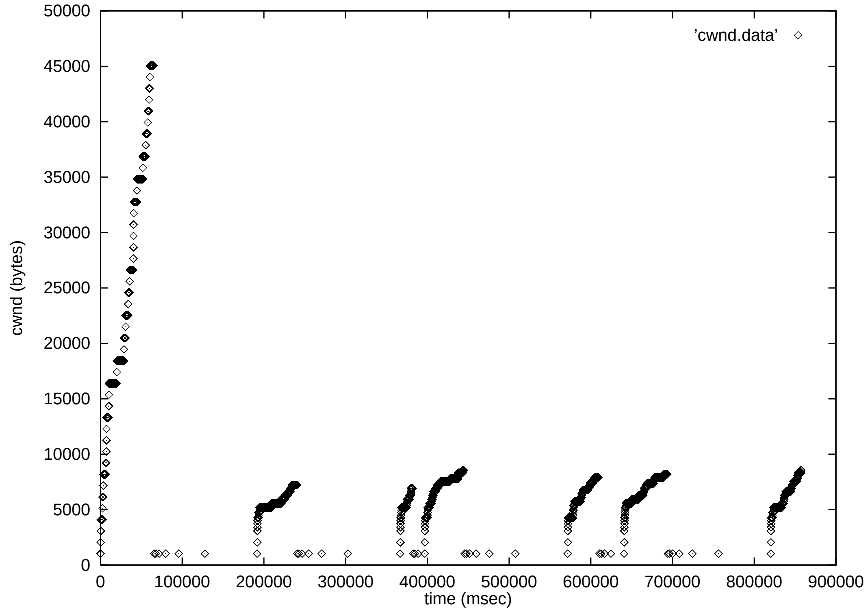

Figure 4.6: Behaviour of the Congestion Window (Medium Range)

In order to observe end-to-end TCP performance when the wireless link is stressed, the load on the wireless link was increased by adding a second connection to run simultaneously between the sender and the receiver. This also increases the load on the sender and the receiver. Table 4.2 shows that for close range, the number of packets lost with light load is similar to the number of packets lost with heavy load, as seen from the number of retransmissions. But there is a decrease in throughput because of actions of the flow control mechanism. In this case the receiver is unable to keep up with the sender. This is seen in Table 4.3 as an increase in the number of zero window advertisements when compared to the light load. Whenever the sender receives a zero window advertisement it stops sending data, until it receives a window update. It is possible that these updates may be lost and the sender probes the receiver. It is seen from Table 4.3 that for the medium range, the sender sends such zero window probes.

Table 4.2: Results From the Heavy Load

Range	Retransmissions	Completion Time (seconds)		Throughput Mbits/sec	
	Average	Average	Std.dev	Average	Std.dev
Close (approx. 0.25m)	4.0	776.85	44.56	0.102	0.005
Medium (approx. 15m)	63.3	1363.32	254.59	0.061	0.012

Table 4.3: Windowing Behaviour

Range	Zero window advertisements		Zero window probes	
	Light Load	Heavy Load	Light Load	Heavy Load
Close (approx. 0.25m)	24.0	108.8	0	0
Medium (approx. 15m)	52.6	117.6	0	12

4.2 Conclusions

There is significant performance degradation of TCP in the presence of a lossy wireless link, since TCP is unable to distinguish between packet loss resulting from errors on the wireless link and packet loss due to congestion on the network. This has the following undesirable consequences:

- The sender waits a longer time to retransmit the lost packets instead of trying to recover from the loss more immediately.
- The sender slows down the rate of transmission of data to prevent "further congestion".

Since the TCP sender assumes that the packet loss is due to congestion along the route, it slows down its transmission rate by doubling the retransmission timeout and decreasing the transmission window. In fact, TCP's response may exacerbate the problem rather than alleviate it.

An explicit notification can be sent by the receiver to the sender indicating the presence of the wireless link. This information can be used by the sender to prevent invoking the congestion control mechanism. Such an approach would require modifications to all the fixed hosts on the Internet. This also means that the sender is becoming aware of the mode of access of the receiver. The sender should not care about how the receiver is accessing the network- through an Ethernet, a dialup connection, a wireless link, or atm.

In a wireless environment, TCP will encounter packet losses that may be unrelated to congestion; however, these losses trigger congestion control measures at the source to the detriment of end-to-end performance. In order for the most popular applications to migrate to the wireless internetworking environment, TCP should be retained. So, while it is necessary to speed up the recovery process, it is impractical to make major modifications to TCP. A possible solution to these problems experienced by TCP is investigated in the next chapter.

Chapter 5

Simulation Models

Previous work in the literature [3, 5] suggests that the use of a proxy (as described in Section 3.9) can provide a viable solution to the problems experienced by TCP in a lossy environment. However, it is not clear yet, how well a proxy performs and in what circumstances it is reasonable to deploy a proxy. Since the source code for the system used in the measurement study was unavailable, it was not feasible to add the proxy module to the wireless hub and measure its impact. This motivated the need for a simulation study to evaluate the performance of the proxy at the base station/wireless hub. Simulation also provides better control over parameters such as the error rates on wireless links, thereby making it easier to study the effects of varying such parameters.

To perform the simulation study, the entire system is represented by simple models. The design approach was to keep the components of each model as simple as possible without sacrificing any functionality relevant to this study. The system model used in the simulation study is described in the following section.

5.1 System Model

The system model represents the network configuration used in the measurement study and includes models for the network, the TCP connection, the proxy, and the losses on the wireless segment. The network model includes communication components that are modelled as nodes and links. The TCP connection is modelled

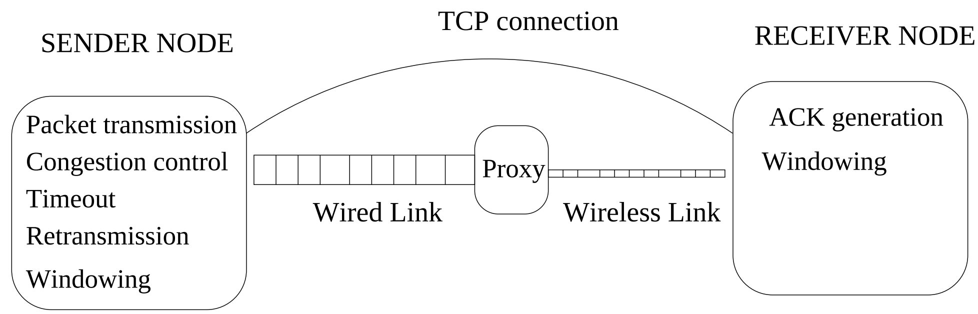

Figure 5.1: Simulation Model

by TCP agents located at the nodes. The end-to-end TCP dynamics of a connection are simulated by these agents. Packet loss on the wireless link is simulated by the loss model and the proxy model simulates the functionality of the proxy mechanism.

There are two types of nodes in the simulated network: endpoints and intermediate nodes. Only endpoints can act as a source/destination of data traffic. Intermediate nodes forward the data packets on to the adjoining links. The proxy is located at the intermediate node. There are two types of TCP agents: source agents and sink agents. The mechanisms for the end-to-end TCP dynamics of a connection are simulated by these source and sink agents. Source agents have mechanisms for sending TCP packets and for receiving ACKs, for packet retransmission (including Karn's algorithm), for RTT estimation, and for congestion control. Sink agents have mechanisms for receiving packets and for generating cumulative ACKs, and a reassembly queue for storing out-of-order packets. These agents are associated with the nodes at the endpoints.

The node associated with a TCP source agent functions as the sender and the node associated with a TCP sink agent functions as the receiver. Hence, the sender can generate packets and receive acknowledegments while the receiver can receive packets and generate ACKs. Both the sender and the receiver can also forward packets on to the links to which they are attached. The intermediate node can only forward the packets onto the outgoing link: it can neither generate nor receive packets since no TCP agent is associated with the intermediate node.

The nodes in the network are connected by links. Thus a link represents the ability of two nodes to communicate directly with each other. Each link is duplex: packets can be flowing in both directions at the same time. A duplex link is simulated between two nodes by using two links, an outgoing link and an incoming link. It is assumed that there is only one route between two nodes. Each link is modelled as a queue and packets are enqueued and dequeued at given time instances. Whenever a packet is being transferred on the link, the link is marked "busy". Packets are enqueued and then forwarded on the link when the link is not busy. Hence, the packets have to wait in this queue when the link is busy. There are two types of links: wired links and wireless links. Both are characterized by bandwidth and latency. In addition, the wireless links also experience packet loss that is simulated using loss models as described in Section 5.2. It is assumed that there is no loss on a wired link.

The data source is an infinite data source that can send data throughout the duration of the simulation. Data is generated from the source in response to ACKs received from the sink. This implies that the source always tries to fill the pipe in accordance with the self-clocking behaviour as described in Appendix A. The packet size is constant throughout the simulation and there is no segmentation of packets while the packets are being transferred over the connection.

5.2 Loss Models

In a network, packets can be lost because of failure of network equipment, buffer overflow due to congestion, or the lossy nature of the link. Packet loss due to the failure of network equipment is rare and is not considered in this analysis. For this study, it is assumed that there is no congestion in the network. In this way, loss due to the wireless link can be isolated from loss due to congestion in the network. It is assumed that the end-to-end effects observed in the experiments result only from loss on the wireless link. The design of the loss model is explained in the following paragraphs.

As a packet is transmitted over a physical link, bit errors may be introduced into the data packet, i.e. a 1 is turned into a 0 or vice versa. Sometimes single bits are corrupted, but normally a burst error occurs, causing corruption of several bits. These errors are usually detected and corrected by the link level protocols and are transparent to the transport layer. If these bit errors cannot be corrected by the link level protocols the entire packet is discarded. Such errors are more common in a wireless link than in a wired link thereby causing loss of entire TCP packets. It is not uncommon to have a link failure in the case of wireless links. In this study, packet loss is modelled using packet error rates rather than bit error rates. In this way, even the loss is abstracted towards the transport layer.

Since the objective is to evaluate TCP performance over a lossy wireless link, it is necessary to incorporate an error model into the simulator. The wireless link has a higher error rate than the wired link and the lossy nature of the wireless link can be captured using a loss model. Losses on the wireless link can be due to various factors, such as the distance between the transmitter and receiver, the power of the antennas, interference in the frequency ranges, fading, or reflections from obstructions. It is difficult to capture all these different losses in a single model but it is important to note that whatever the reason for loss on the wireless link, it is perceived as a loss of packets by the higher layer protocols. Also, since the objective of this work is to assess the behaviour of a higher layer protocol in reaction to loss on the wireless link, the emphasis is on protocol behaviour rather than on the nature of the loss. For this reason, a detailed loss model was deemed not to be necessary. The idea was to keep the loss model simple and yet be able to capture the packet loss behaviour and the end-to-end effects due to this packet loss. The error models described in this section do not capture individual losses but only the loss due to which an entire packet is lost.

Three simple loss models were used in the simulation experiments: a deterministic loss model, a random loss model, and a bursty loss model. In the deterministic loss model, packets on the wireless link are dropped in a periodic fashion. A 10% packet error rate implies that 1 out of every 10 packets is lost. In the random loss model, packets are dropped randomly using a Poisson distribution. In such a model, an average of 1 packet is dropped for every N packets. Varying the value of N causes different error rates. A packet loss of 10% indicates that approximately 10% of the total packets are dropped at random instances in the simulation. However, there are also occasions when multiple packets are lost in each window. Yavatkar and Bhagawat [59] used such models to evaluate the performance of their TCP implementation. Balakrishnan *et al.* [5] used a similar model with a bit error rate of 10^{-6} to demonstrate the effectiveness of the Snoop protocol.

Nguyen *et al.* [41] have found empirically that the loss characteristics of a wireless link are bursty in nature, due to the combination of various effects. They proposed a sophisticated Markov model based on traces obtained from a wireless LAN. Other studies have also established that finite state Markov models can be effectively used to characterize bit error patterns observed on RF channels [58, 52].

A two-state Markov model was used to capture the burstiness of loss on the wireless link. Burstiness of loss means that, although the wireless links are usually well behaved, they occasionally go into a bad state in which many burst errors occur within a small time interval. Similar models have been used by other researchers [7, 18]. The model consists of two states as shown in Figure 5.2: an OFF state and an ON state. At any particular instant during the simulation, the wireless link is in either the ON state or the OFF state. Three packet error characteristics can be used to describe the burstiness:

- error rate
- error length (the number of packets that are lost consecutively)
- error-free length (the number of packets that are successfully received between two adjacent bursts of error)

Each of the states has its own distribution. It can be assumed that the times spent in the ON and OFF states are exponentially distributed, but with different average values.

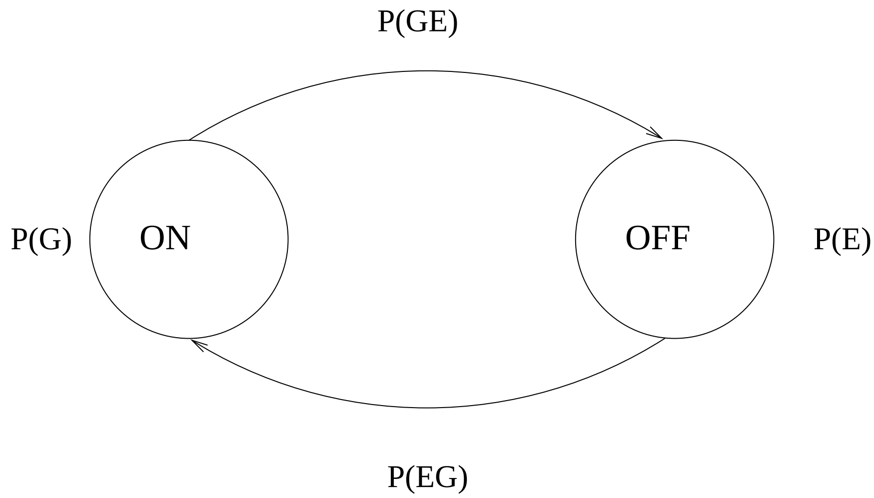

Figure 5.2: A Two-State Markov Model

P(E) denotes the probability in the OFF state and P(G) denotes the probability of remaining in the ON state. P(EG) denotes the transitional probability from the OFF to the ON state and P(GE) denotes the transitional probability from the ON state to the OFF state. The error length and error-free lengths are translated to an error duration and interarrival time between bursts, respectively. "Mean ON time" refers to the time that a wireless link is in the ON state and "Mean OFF time" refers to the time that the wireless link is in the OFF state. When a wireless link is in the OFF state, all packets that are to be transmitted over the link are dropped. In the ON state, all the packets are successfully received. Hence, the time duration in the OFF state determines the burstiness of errors. The values of these two parameters remain constant throughout the duration of the simulation.

Initially, at the start of the simulation, the wireless link is in the ON state. After the mean ON time, there is a transition from the ON state to the OFF state. Again after the mean OFF time there is a transition from the OFF state to the ON state. Such transitions take place throughout the simulation. So, by varying the mean periods spent in the ON and OFF states, different error rates can be modelled.

The above three loss models are integrated into the simulator. For a particular set of experiments, one of these loss models was used to simulate different error rates on the wireless link. According to the distributions obtained from these loss models, packets are dropped for certain intervals of simulated time. While the exact performance numbers are strongly dependent on the error model, the relative performance numbers are dependent on how the protocol behaves after one or more losses in a single TCP window.

5.3 Modelling the Round-Trip Time of the Connection

It is important to model packet propagation through the network. The round-trip time (RTT) of a TCP packet is the time elapsed between its release from the sender and the reception of the corresponding ACK by the sender. This time includes processing delays at the source, sink, and intermediate nodes in addition to the propagation and queueing delays for the packet and the ACK.

Each TCP packet is transferred in an IP datagram and each IP datagram is transferred in an Ethernet frame. In a real system, each Ethernet frame is further divided into wireless frames based on the link layer protocol on the wireless link. The IEEE 802.11 protocol breaks down each Ethernet frame into 128 byte wireless frames. These wireless frames are transmitted on the wireless link. IEEE 802.11 has its own retransmission mechanism and if one of these frames is corrupted or lost, this frame is retransmitted from the wireless hub to the receiver. If any of these frames are lost the entire TCP packet has to be retransmitted from the sender to the receiver.

The parameter "processing delay" is used to represent delays due to protocol processing at the sender and receiver. In addition to the transport layer processing overhead, there are delays associated with protocol processing at the lower layers, at both the sender and receiver. There are additional processing delays at the wireless hub. Therefore the total processing delay = processing delay (wired) + processing delay (wireless). The processing delay values remain constant throughout the duration of the simulation. This means that such effects contribute equally to all the experiments. Thus, in a way these effects are isolated and the only factor varying between experiments is the loss on the wireless link.

There are three factors that determine the time to send a packet across a link: the propagation delay, the transmission delay and the queueing delay. Because of the finite speed of light, there is a very small amount of delay while transferring even a single bit across a link over a certain distance. Since the distance between the hosts in these experiments is very small, this delay is negligible and is not considered further. Transmission delay refers to the time taken to transfer a unit of data. This indicates the number of bits per second that the medium can consecutively transmit. Transmission delay is a function of the network bandwidth and the size of the data packet. Thus for a given path between two nodes, transmission delay can vary with the packet size. The transmission delay on each of these links is therefore expressed as (packet size / bandwidth). Since bandwidth of the wired link is different from the bandwidth of the wireless link, the transmission delay differs for the wired and wireless links. The total transmission delay seen by the TCP agents is the sum of the delays experienced over the wired and wireless links. For a given path between two nodes, propagation delay is fixed while transmission delay depends on the packet size. Packets are usually queued at switches or routers before being forwarded on to the outgoing links, causing queueing delays inside the network.

In the simulation model, the packets are actually queued at a node and transmitted on a link when the link is not busy. Thus the queueing delay is also represented in the model. RTT dynamically varies throughout the duration of the simulation because the queueing delay varies for different packets and is captured in the simulation model.

Latency is expressed as the sum of the above three delays.

Latency = propagation delay + transmission delay + queueing delay

Total Latency = Latency (wired) + Latency (wireless)

The RTT is the sum of the latency of packets in one direction and the latency of ACKs in the opposite direction.

RTT = Total Latency of the packet + Total Latency of the corresponding ACK

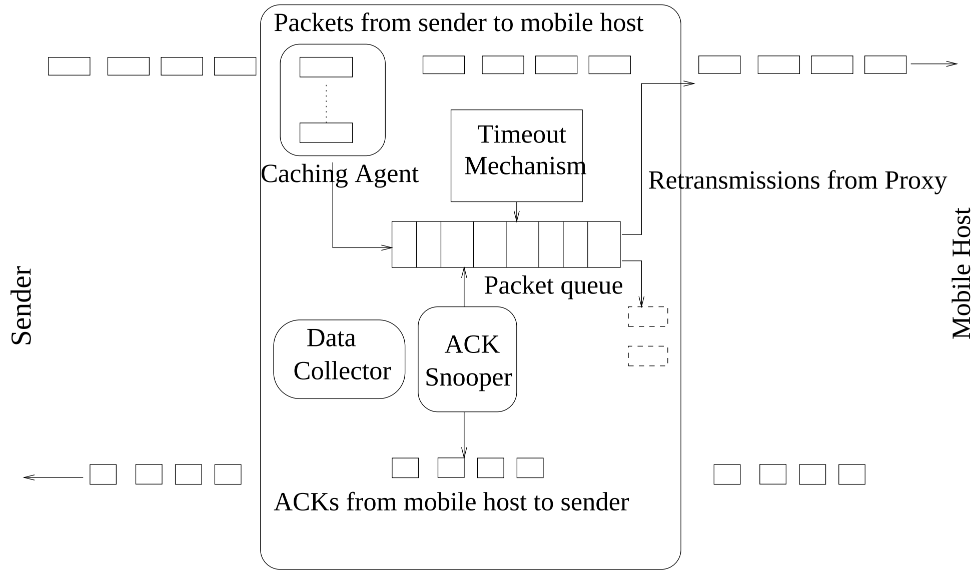

Figure 5.3: Proxy Model

5.4 Proxy Model

In this study, the proxy performs packet retransmissions on behalf of the sender. The proxy is transparent to the sender, ie. the sender is unaware of the presence of the proxy. The proxy is located at the intermediate node that behaves as the wireless hub/base station. By performing local retransmissions on the wireless link, the goal of the proxy mechanism is to try to avoid timeouts at the sender, thereby preventing the invocation of the congestion control mechanism at the sender.

A model of the proxy employed is shown in Figure 5.3. The proxy stores certain state information about a connection. It keeps track of the packet sequence numbers, the size of the data packets and the IP addresses and port numbers of the endpoints. Packets flowing from the sender to the receiver are cached at the proxy before being transmitted over the wireless link. In this model a queue is used to cache these packets.[1] When ACKs are flowing from the receiver to the sender through the intermediate node, the packets with corresponding sequence numbers are flushed

[1]This queue needs only to be large enough to handle the maximum window size of the TCP connection since at any instant the total number of packets flowing over a TCP connection is limited by the maximum window size. This assumption is valid, however, only for a single TCP connection.

out of the queue by the ACK snooping mechanism. Since the ACKs are cumulative, more than one packet may have to be flushed out. Data packets are retransmitted from the proxy if the corresponding ACKs do not flow through the proxy within a timeout interval. When packets are retransmitted from the proxy because of such timeouts at the proxy, the receiver generates an ACK. The receiver cannot distinguish between packets coming from the actual sender and packets coming from the proxy. The value of this timeout is estimated dynamically, based on an algorithm similar to RTT estimation of a TCP connection. Whenever a packet is cached at the proxy, the packet is also time-stamped, and when the ACK is received the round-trip time on the wireless link is calculated. A running average of this value is maintained at the proxy for all the packets throughout the simulation.

An important design consideration for the retransmission mechanism at the proxy is that it should not retransmit packets without any packet loss on the wireless link. This sets a lower bound on the initial timeout value at the proxy. If the initial timeout value at the proxy is lower than this lower bound, the proxy retransmits packets that are not actually lost. This means that although the original packets from the sender have been successfully received, the timeout value at the proxy is so small that the timer expired before the ACKs flow through the proxy. Unlike the retransmission mechanism at the TCP source agent, the timeout mechanism at the proxy does not use any backoff mechanism. The design makes use of the knowledge that the loss observed is due to the wireless link and not due to congestion. This makes the retransmission scheme at the proxy more aggressive. It is to be determined experimentally whether such a policy speeds up the recovery process. The processing delay at the proxy is negligible when compared to the overall round-trip time of the connection and hence is not modelled separately. In a real network, this delay is captured in the initial RTT estimation and is not large enough to affect the end-to-end throughput in a significant fashion.

Figure 5.4: Configuration to Test TCP Functionality

5.5 Validation of the Model

To have any confidence in the results obtained, the models used in the simulation study must be reasonable approximations to the real network. Validation ensures that the assumptions used in developing these models are reasonable [25]. This implies that if correctly implemented, the model should produce results close to that observed in a real system.

To validate a simulation model three major points must be established:

- all relevant components are modelled
- the components of the simulator function properly
- the output values are reasonable.

These issues can be established by demonstrating that the behaviour of the simulation is consistent with functional specification and by comparing results from simulation with measurements from the real system.

Two models need to be validated for this investigation.

- the model of TCP
- the model of packet loss over the wireless link.

Establishing the functional correctness of the TCP model involves determining that the various components of the model function as they are supposed to – in particular, that the various components function as specified in RFC 793 [46]. The TCP model was first validated independent of any wireless component by running the simulator without the proxy and without the wireless link. In this case, the network configuration consists simply of a sender and a receiver connected by a

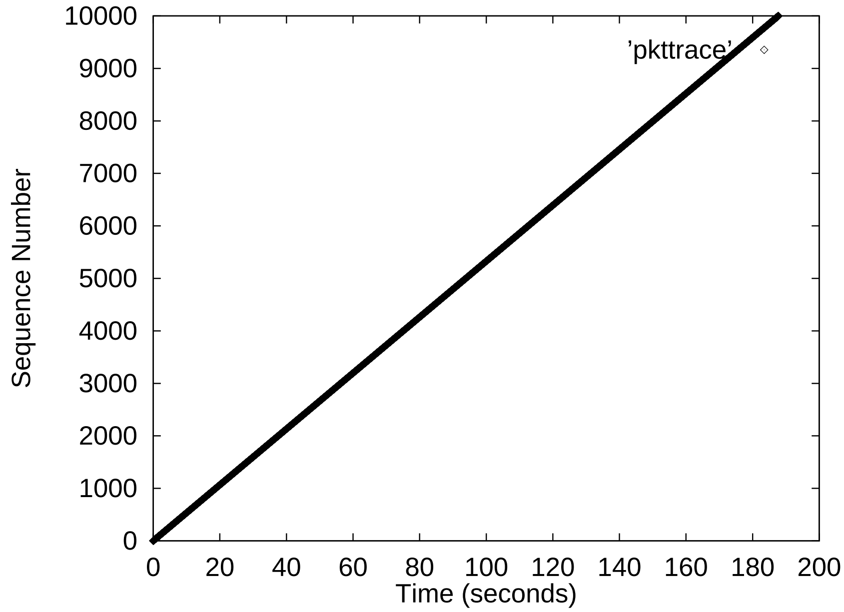

Figure 5.5: Packet Trace: No Loss

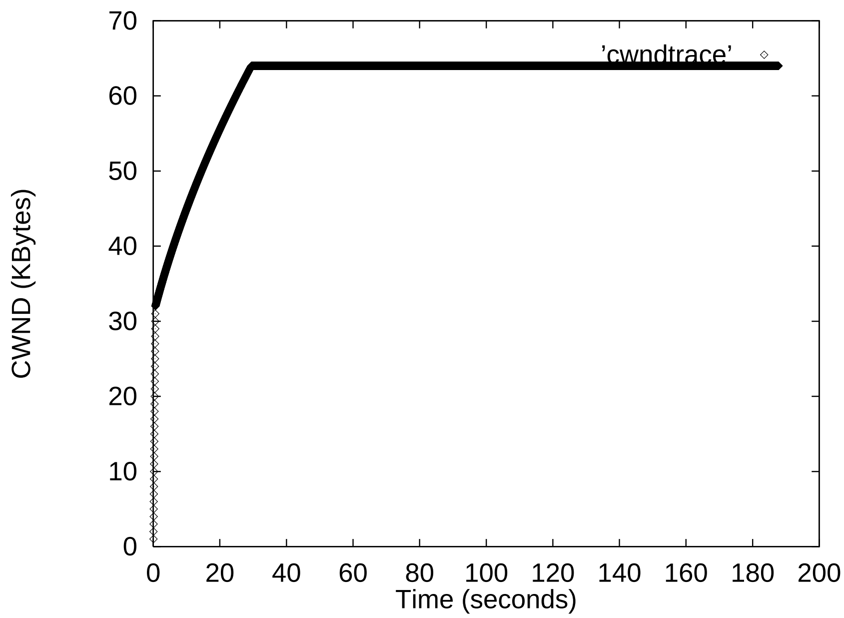

Figure 5.6: Behaviour of the Congestion Window: No Loss

link of 10 Mbps bandwidth as shown in Figure 5.4. TCP packets were generated by the sender and acknowledgements were generated by the receiver. A trace of sequence numbers of packets and acknowledegments was obtained, and this trace was used to verify that the acknowledgement scheme performs in accordance with the specification.

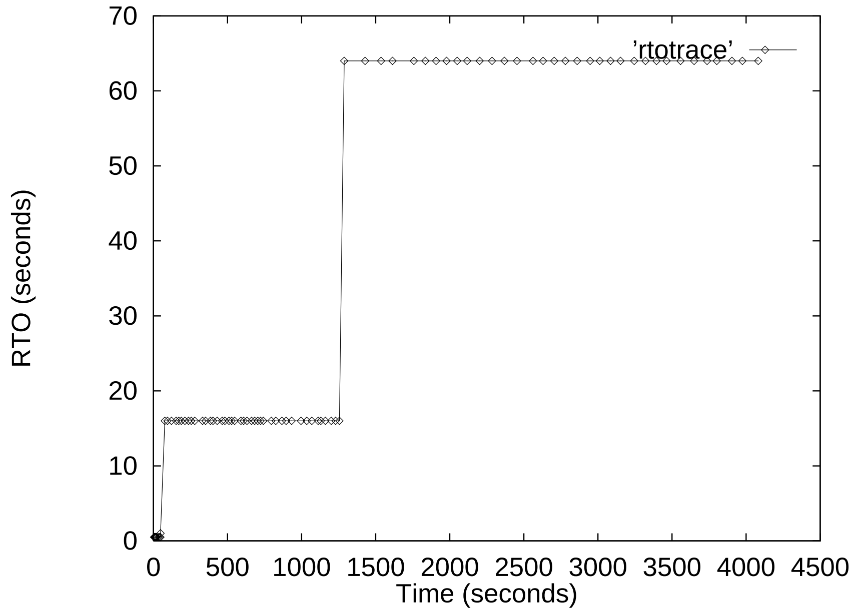

Figure 5.7: Behaviour of RTO: Lossy Link

From the traces of the packet sequence numbers and ACKs, it was determined that acknowledgements are not generated for packets that haven't been transmitted. It is seen from Figure 5.5 that there is no packet loss when the loss models are not used in the simulator. This shows that the packet generation mechanism functions properly.

Observations of the congestion window (cwnd) parameter were used to determine that the congestion control scheme works correctly. The behaviour of the congestion window mechanism as shown in Figure 5.6 is also according to specification. It was also observed there are no timeouts at the sender and hence there is no backoff of the RTO timer.

To verify the retransmission scheme, packet loss was introduced. This was accomplished by introducing a wireless link between the sender and receiver and observing the behaviour of the retransmit timeout value (RTO) throughout the duration of the simulation. Figures 5.7 and 5.8 show that the timeout and retransmission mechanisms function correctly. In Figure 5.8 it is seen that the congestion control algorithm is in the exponential phase until the threshold value is reached and then switches to the linear phase. Whenever there is loss, cwnd is reinitialized to 1 and

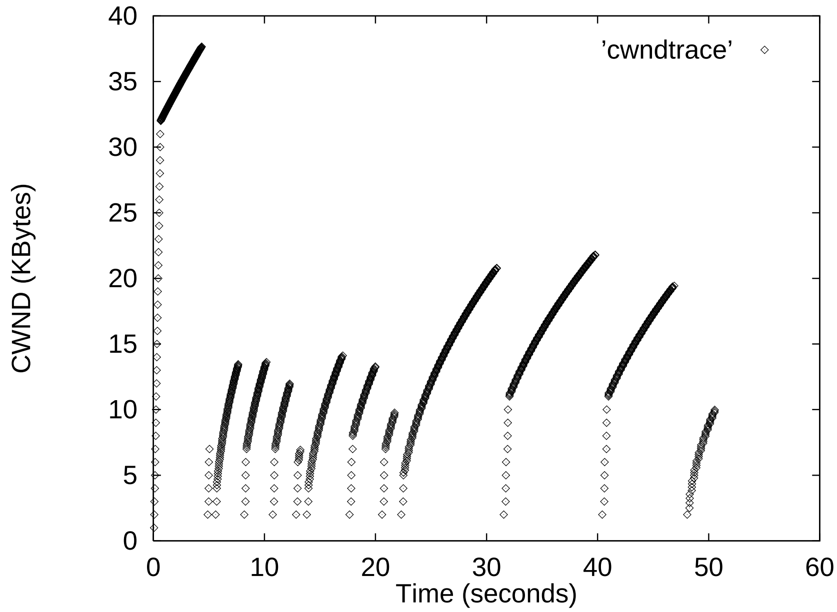

Figure 5.8: Behaviour of the Congestion Window: Lossy Link

the threshold value is updated to half of the congestion window size.

The loss model was validated in conjunction with the TCP model by establishing that the model is able to reproduce packet retransmission behaviour similar to that observed in the measurement study. The simulator was run without the proxy and end-to-end measures such as throughput, number of retransmissions, and timeouts were used to compare results from simulation with the results obtained empirically. This indicates to what extent the model is able to capture the end-to-end effects of the actual system. The differences between the values obtained through simulation and the values obtained through measurement indicate the deviation of the model from the actual system. If *Vsim* and *Vmes* represent the values obtained from simulation and measurement respectively, then *(Vsim - Vmes) / Vmes* gives the relative error. The initial goal was to obtain less than 10% relative error.

Figure 5.9 shows the throughput and retransmissions obtained from the measurement study and from the system model. At very low error rates (when the number of retransmissions is very small), the throughput obtained from the simulation model is much higher than was measured. This is because, at very low error rates (few retransmissions), the processing delay and the transmission delay are the

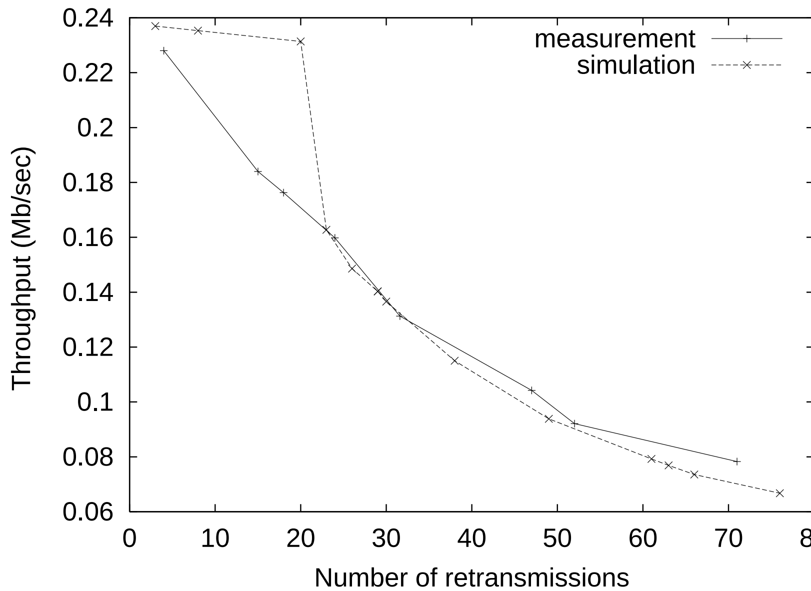

Figure 5.9: Comparison of Values from Measurement and Simulation

dominating factors contributing to the throughput. As mentioned earlier, the processing delay is calculated approximately. At certain error rates, the throughput and the number of retransmissions for the measurement study and the simulation model are very similar. This is because at these error rates, the TCP mechanisms affect the throughput to a greater extent. This also justifies the design decisions in this study: to abstract away as much as possible towards the transport layer and to focus on the end-to-end TCP mechanisms. At higher error rates (many retransmissions), the simulated values once again deviate from the measured values. This can be explained by the fact that the values obtained at those error rates also have very high variance in the measurement study.

Although the output values obtained while using a proxy cannot be validated against measured values (since there are no measurement values with a proxy), the functional correctness of the proxy model can be verified to a certain extent. To verify the functional correctness of the proxy, the proxy model was switched ON and there was no loss on the wireless link. It was observed that the number of timeouts at the sender was still zero and there are no retransmissions from the proxy. It was also observed that there was no change in the end-to-end throughput obtained over the connection. This further demonstrates that the proxy is transparent to the sender.

Validation is an iterative process in the sense that inaccuracies in the initial model might require change to the model. After making these changes, the model must be validated again. The validation helps increase the degree of confidence in the model results. In this investigation, the validation was limited to a few selected scenarios, but all important cases were covered.

For similar configurations, end-to-end throughput and RTT from the measurement study agree with those from simulation. For different losses on the wireless link, the throughput and number of retransmissions obtained in the simulation model were similar to those obtained in the real environment with a similar number of retransmissions. In other words, the simulation model behaves in accordance with the real system.

5.6 Limitations of the Model

Although the model is representative of the system under study, it is different from the real network in several ways. There is much more to TCP's functionality than is modelled in this study, including connection setup and various other timers. This is not a major limitation, however, because the essential dynamics relevant to this study are modelled. The link-level protocol on the wireless link is not modelled. More accurate wireless loss models would be necessary for a thorough evaluation of the wireless environment. Since the focus of this study is not in modelling these losses this was not seen to be necessary. To get more accurate performance numbers, sophisticated wireless loss models would be required to model the losses on the wireless links. Since, the focus of this study is on the end-to-end behaviour and not on any particular wireless environment, it was felt that the choice of simple loss models was justified.

5.7 Summary

This chapter described the various models used in this study. Although the system model is a simplified representation of the actual system, enough detail has been incorporated to provide a good approximation of the real system behaviour. Error models were incorporated into the simulator that allow sufficient control over errors on the wireless link. The system model was validated against the system used in the measurement study. It is assumed that a suitable wireless medium exists and the characteristics of any particular medium are not modelled. Only the packet errors resulting from this link are modelled and the model is abstracted away from the physical layer towards the transport layer as much as possible. It was shown that the TCP model performs correctly both without loss and in the presence of loss. It was also shown that the loss model used replicates the loss behaviour in the measurement study.

The results of the validation show that the system model captures the essentials dynamics of TCP relevant to this study. The results obtained from a detailed performance evaluation of the proxy are presented in the following chapter.

Chapter 6

Simulation Study

6.1 Goals of the Simulation Study

The primary goal of this simulation study is to evaluate the impact of a proxy on the end-to-end performance of a TCP connection in a wireless environment. In particular, the study includes a performance evaluation of the proxy and determines how the proxy affects the behaviour of various TCP mechanisms at different error rates.

6.2 Metrics

In addition to the metrics presented in the measurement study, some additional metrics are proposed to evaluate the performance of the proxy. These metrics demonstrate the effectiveness of the proxy and help in understanding how the proxy affects the various TCP mechanisms.

- **Gain** is defined as an improvement in throughput in the presence of the proxy. This is the major performance metric for the proxy. If Th is the throughput without a proxy and Th(proxy) is the throughput in the presence of a proxy, then the gain for a given error rate = Th (proxy) / Th. A gain less than 1 implies that the proxy is deteriorating the overall end-to-end performance.

- **Number of retransmissions from the proxy**. When original packets from the sender are lost on the wireless link, the proxy retransmits them. Even these retransmissions from the proxy might be lost, causing additional retransmissions from the proxy. This number should be zero when there is no loss on the wireless link or otherwise it implies that the retransmission algorithm at the proxy is not properly implemented. The comparison of the number of retransmissions from the proxy with the number of retransmissions from the original sender also gives a good indication of the effectiveness of the proxy.

- **Number of redundant retransmissions from the sender**. Even in the presence of a proxy, timeouts at the sender cannot be completely avoided. Retransmissions from the sender are still possible for two reasons: retransmissions from the proxy to the receiver can be lost, or ACKs generated at the receiver may not reach the sender quickly enough to prevent a timeout. In the case when the retransmissions from the proxy are lost, the sender has to retransmit them eventually. In the second case, packets have already reached the receiver but still the sender is retransmitting them. These are redundant retransmissions that can be avoided. If the number of such retransmissions is high, this implies that the proxy is less effective. The number of redundant retransmissions from the sender as a percentage of the total retransmissions also gives an indication of the performance of the retransmission mechanism at the proxy.

- **Number of timeouts at the sender** The number of timeouts at the sender can also be used to judge the effectiveness of the proxy. Once again these values can be compared between the experiments in which the proxy is ON and the ones in which the proxy is OFF. Retransmissions from the proxy are expected to generate acknowledgments from the receiver, thereby avoiding a timeout at the sender. Preventing a timeout at the sender prevents the congestion control algorithms from setting in at the sender, resulting in increased throughput. The number of such timeouts in the presence of high error rates will indicate whether or not the proxy is useful, even in the presence of high error rates. The maximum retransmit timeout (RTO) value at the sender also helps in

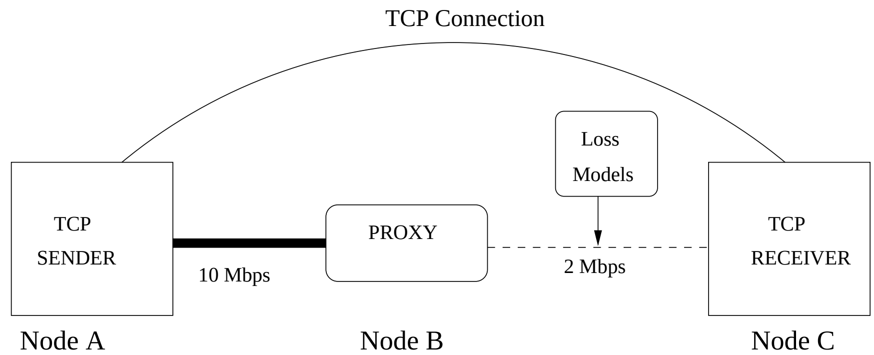

Figure 6.1: Network Topology for the Experiments

this evaluation. The maximum RTO value reached for the connection gives an indication of the extent of backoff during the data transfer.

- **Throughput vs error rate** The variation of the throughput with the error rate on the wireless link is an important performance consideration. For a given error rate, the throughput obtained from the experiments when the proxy is ON can be compared with the throughput obtained when the proxy is OFF. It is expected that throughput improves when the proxy is ON. It will be interesting to see if there is a significant increase, particularly in the presence of high error rates.

6.3 Design of Experiments

The design involves setting up the simulator configuration for various experiments, and selecting the parameters/factors to be varied during these experiments and selecting the values/levels through which they are varied. The network topology and the various simulation parameters are described in the following subsections.

6.3.1 Network Topology

The network topology is similar to the topology in the measurement study. There are three nodes and two links as shown in Figure 6.1. One of the nodes (Node A) is associated with the TCP source model and acts as a sender of data. Node C is associated with the TCP sink model and acts as a receiver. The proxy model is associated with the intermediate node, Node B and acts as base station, performing local retransmissions between B and C. A link is created between nodes A and C with a bandwidth of 10 Mb/sec and no loss model is associated with it. This link behaves as the wired link. A link with a bandwidth of 2 Mb/sec is created between nodes C and B and a given loss model is associated with it. This link behaves as the wireless link. The routing of packets between these nodes is done by simple forwarding. The details of these individual models were described in the previous chapter.

As data is transferred between the source and the sink over the TCP connection, the trace module collects the data for the performance analysis. This includes end-to-end characteristics such as completion time, throughput, and number of retransmissions, and the values of certain variables such as congestion window, retransmit timer and sequence numbers of the packets. After an event is performed in the simulation, all the necessary values are logged into the trace module.

6.3.2 The Workload

The data source is modeled as a bulk data source generating TCP packets throughout the duration of the simulation. Each TCP packet is assumed to be 1024 bytes. The effect of packet size on the throughput of the connection might be an interesting parameter to investigate but this does not affect the proxy mechanism. Hence, the TCP packet size was held constant during the experiments. Each simulation runs until a total of 10,000 packets are sent.

6.3.3 Experimental Methodology

The experimental suite for the simulations is shown in Figure 6.2. Experiments are conducted with the proxy OFF or ON. When the proxy is ON, the proxy model performs local retransmissions on the wireless link. When the proxy is OFF, the functionality is similar to that of an intermediate node in the network: it simply

	Proxy Off	Proxy On
Deterministic Loss	I	II
Random Loss	III	IV
Bursty Loss	V	VI

Figure 6.2: Experimental Suite for the Simulations

transfers packets from one link to the other.

The simulator is first set up so that there is no loss on the wireless link and the proxy is switched OFF. This is the "best case" scenario in the presence of a wireless link and the value of throughput is the maximum throughput that can be achieved over this connection. This gives the baseline values for throughput and completion time, values that can be compared with other values obtained in the presence of loss on the wireless link.

6.3.4 Simulation Parameters

Four system parameters are identified for this simulation study: the error rate, the error policy (describing the nature of the errors), the state of the proxy, and the window size.

Error rate is expressed as a percentage and corresponds to the percentage of packets that are lost. The error rate was varied through the following levels: from 0.1% to 1.0% in steps of 0.1, from 1 to 10% in steps of 1, and from 10 to 25% in steps of 5.

Three different error models were used to demonstrate the effectiveness of the

Table 6.1: No Loss: Window Size = 4K

Error Rate (%)	Completion Time (seconds)	Throughput (Mb/sec)	Retransmissions	Max RTO (seconds)
0	244.11	0.336	0	0.5

proxy mechanism in different error scenarios, in particular to determine if the nature of errors has an impact on the throughput of a connection and also on the performance of the proxy.

The proxy can be either in the ON state or in the OFF state. Analysis of results obtained in these two states determines the effectiveness of the proxy.

Initially, experiments were performed without any loss on the wireless link. After obtaining these no-loss values, the proxy was OFF and a Poisson error model was used to simulate errors on the wireless link. The error rate was varied for each experiment. These were followed by another set of experiments in which the proxy was switched ON and the error rates were varied in a similar fashion. Similar sets of experiments were also conducted for the deterministic error model and the bursty error model.

6.4 Results and Discussion

Table 6.1 shows the completion time of the simulation (transmission of 10,000 packets) and the throughput of the connection when there is no loss on the wireless link. The throughput represents the maximum throughput that can be achieved for this configuration with a window size of 4 KBytes. Since no loss was experienced by the connection, the number of retransmissions from the sender is 0 and there is no backoff for the retransmit timer.

Table 6.2: Random Loss Model: Without Proxy

Error Rate (%)	Completion Time (seconds)	Throughput (Mb/sec)	Retransmissions	Max RTO (seconds)
0.05	345.67	0.23699	3	0.50
0.10	348.13	0.23531	8	0.50
0.20	354.06	0.23138	20	1.00
0.30	599.74	0.13659	30	16.00
0.40	712.28	0.11501	38	16.00
0.50	872.88	0.09385	49	16.00
0.60	1033.75	0.07925	61	16.00
0.70	1113.83	0.07355	66	16.00
0.80	1227.47	0.06674	76	32.00
0.90	3532.27	0.02319	83	64.00
1.00	3949.13	0.02074	93	64.00
2.00	11769.67	0.00696	218	64.00
4.00	25809.57	0.00317	429	64.00
5.00	34288.54	0.00239	544	64.00
6.00	42779.01	0.00191	671	64.00
7.00	51834.86	0.00158	813	64.00
8.00	58939.18	0.00139	925	64.00
9.00	65595.05	0.00125	1029	64.00
10.00	72699.01	0.00113	1140	64.00
15.00	111258.32	0.00074	1743	64.00
20.00	161595.13	0.00051	2525	64.00
25.00	215806.39	0.00038	3372	64.00

Table 6.3: Random Loss Model: With Proxy

Error Rate	Completion Time	Throughput	Retransmissions		Max RTO
(%)	(seconds)	(Mb/sec)	(Proxy)	(sender)	(seconds)
0.05	244.91	0.33450	3	0	0.50
0.10	246.11	0.33286	8	0	0.50
0.20	249.02	0.32897	20	0	0.50
0.30	251.08	0.32628	30	0	0.50
0.40	282.23	0.29026	38	1	1.00
0.50	291.55	0.28098	49	2	2.00
0.60	301.56	0.27165	61	2	4.00
0.70	305.96	0.26775	66	2	4.00
0.80	314.47	0.26050	76	2	8.00
0.90	320.07	0.25594	83	2	8.00
1.00	328.48	0.24939	93	2	8.00
2.00	438.86	0.18667	219	4	16.00
4.00	708.18	0.11568	430	17	32.00
5.00	1213.94	0.06748	545	28	32.00
10.00	6908.47	0.01186	1140	101	64.00
15.00	15271.73	0.00536	1744	223	64.00
20.00	28744.61	0.00285	2525	434	64.00
25.00	43031.69	0.00190	3372	648	64.00

6.4.1 Evaluating Proxy Performance using the Random Loss Model

Table 6.2 shows results obtained when running the experiments with the proxy OFF and using the random loss model to simulate errors on the wireless link. Table 6.3 shows the results obtained with the proxy ON and the random error model. Table 6.3 also shows the number of retransmissions performed by the proxy. The number of local retransmissions from the proxy steadily increases as the error rate increases.

To understand these results better, the throughput achieved with and without the proxy is shown in Figure 6.3. The throughput is always higher with the proxy ON. This is because the number of timeouts at the sender is significantly reduced with the proxy ON as shown in Figure 6.4. In other words, from the sender's point of view, the connection is experiencing a lower error rate. Retransmissions from the proxy generate ACKs from the receiver, resulting in fewer timeouts at the sender. Fewer timeouts implies that the proxy is shielding the sender from the effects of

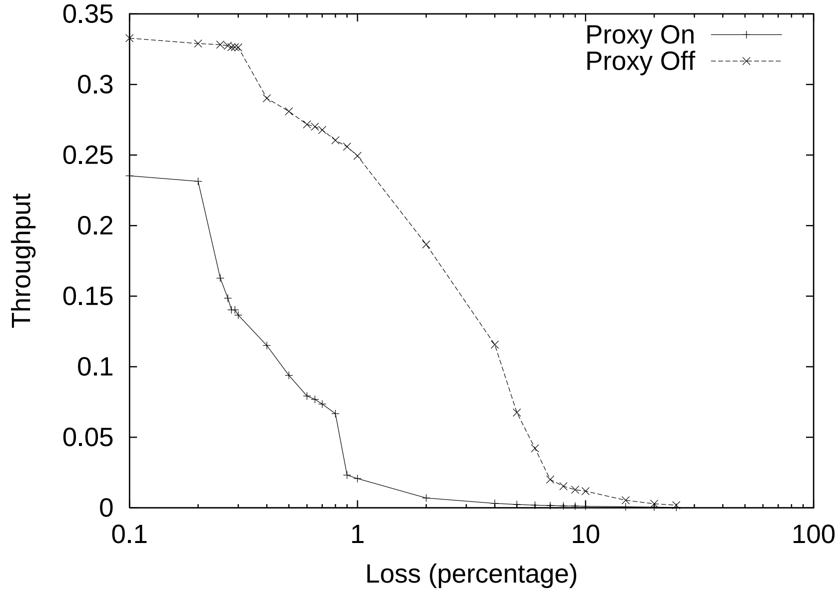

Figure 6.3: Comparison of Throughput with and without the Proxy

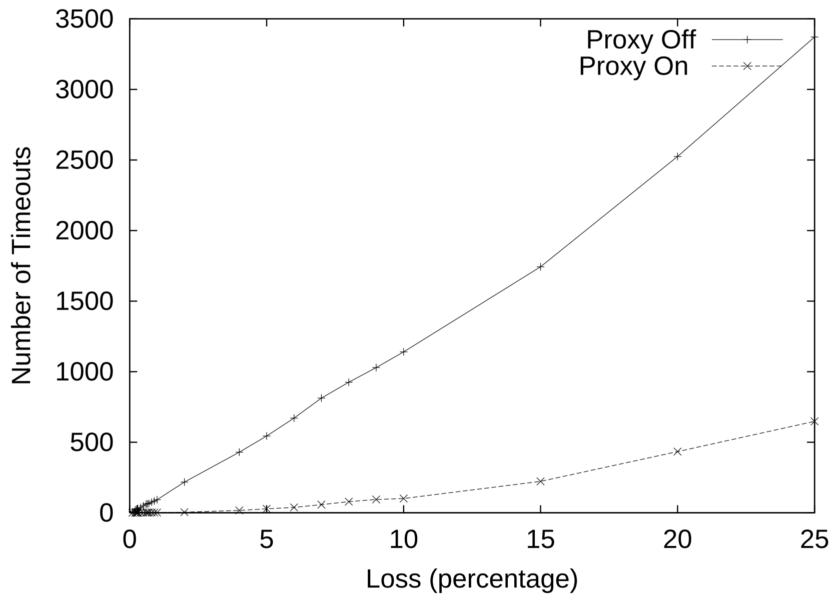

Figure 6.4: Comparison of Timeouts at the Sender with and without the Proxy

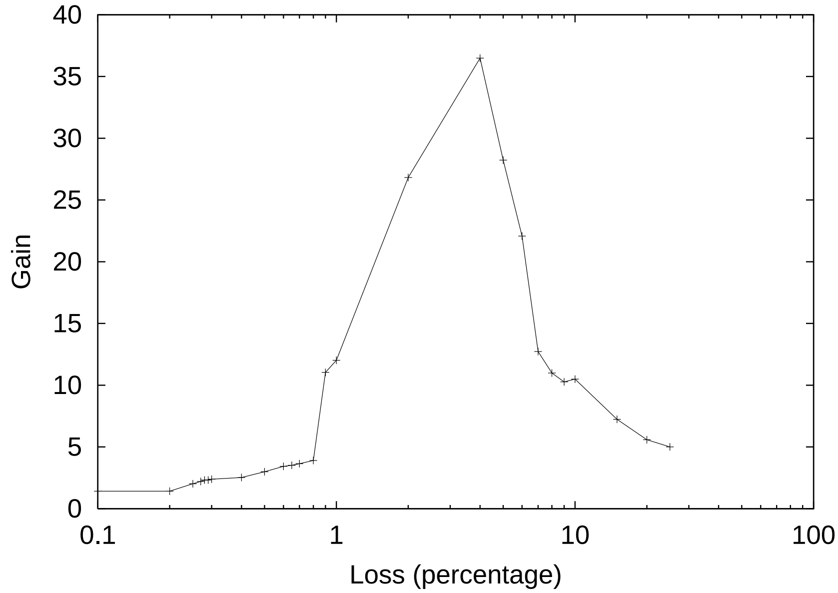

Figure 6.5: Gain in Throughput with the Proxy

losses on the wireless link to a certain extent.

Table 6.3 shows that there are no retransmissions for error rates between 0.1 and 0.3% since there are no timeouts at the sender. Retransmissions at the sender steadily increase for error rates up to 2.0% because the number of timeouts increases. Beyond those error rates, the number of retransmissions from the sender increases significantly. This implies that the proxy could not avoid timeouts at the sender beyond certain error rates, i.e. the proxy is not able to completely mask the effects of loss on the wireless link.

The increase in throughput is further elaborated by Figure 6.5. This figure shows the gain in throughput with a proxy for different error rates. At low error rates (up to 0.8%) the effect of loss is not significant and hence the gain obtained with the proxy is less. At high error rates (beyond 10%), timeouts at the sender are unavoidable even with the proxy, and so the gain is not very high. For error rates in the range of around 5%, when the loss is significant and the proxy is able to prevent the timeouts and thereby the backoff mechanism, the gain is very high.

There is no significant backoff for the low loss ranges, as seen in Table 6.3 and Figure 6.6, and for the high loss range maximum backoff could not be avoided even

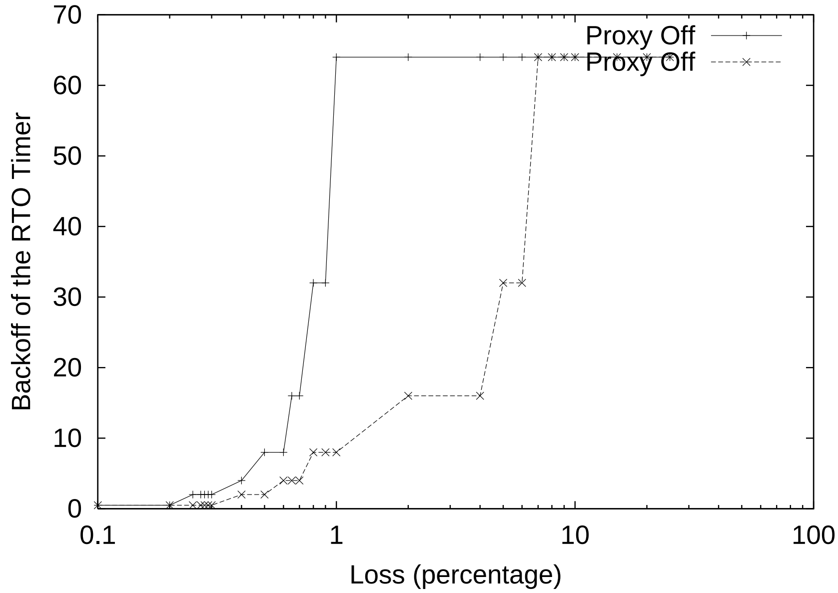

Figure 6.6: Comparison of Timeouts at the Sender with and without the Proxy

with the proxy. This is shown in Figure 6.6. At error rates beyond 5%, it is seen that even in the presence of the proxy the maximum RTO value reached is the same (64 seconds) as in the case without the proxy. However, at error rates around 5% the maximum RTO value reached with the proxy is much less than the value reached without a proxy. This further accounts for the shape of Figure 6.5.

For error rates between 0.1% and 1% the throughput achieved is very high, after which it falls down sharply. The degradation in throughput for error rates beyond 1% can be explained by the fact that the RTO value reaches a maximum of 64 seconds beyond 1% error rates. This means that for error rates beyond 10% the RTO timer has backed off enough to reach the maximum RTO value.

As explained in Section 6.2, redundant retransmissions from the proxy cannot be avoided at high error rates. These redundant retransmissions are shown in Figure 6.8. In Figure 6.9 these redundant retransmissions are expressed as a percentage of the total retransmissions from the sender. An efficient proxy design should seek to minimize these redundant retransmissions at high error rates.

In the following subsections, these results are analyzed in greater detail. In Subsection 6.4.2, the behaviour of various TCP variables is examined for different

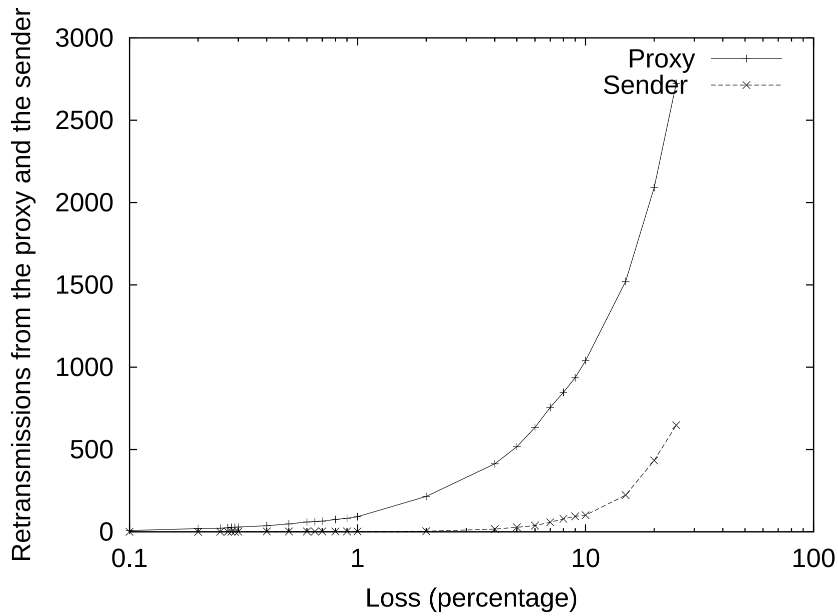

Figure 6.7: Retransmissions from the Proxy

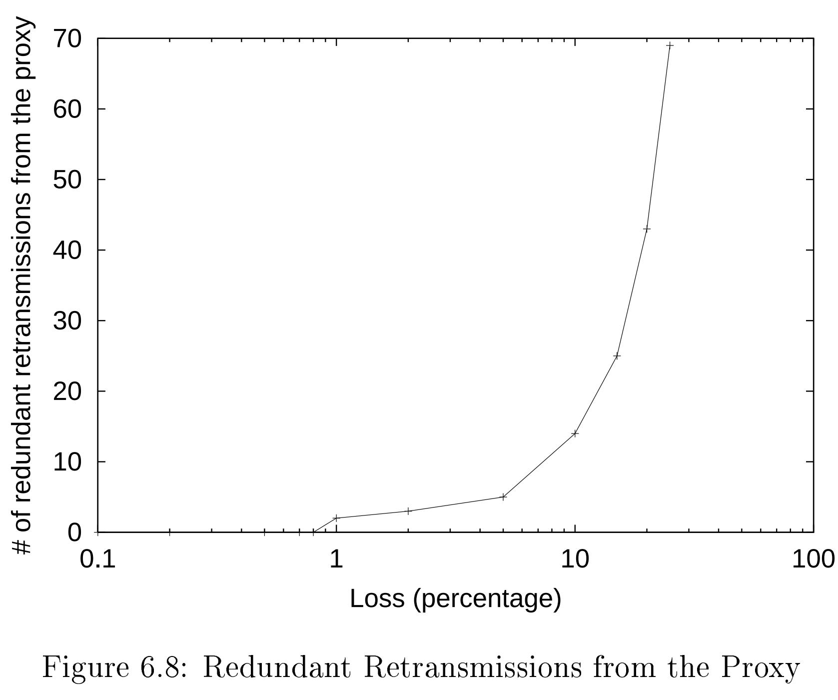

Figure 6.8: Redundant Retransmissions from the Proxy

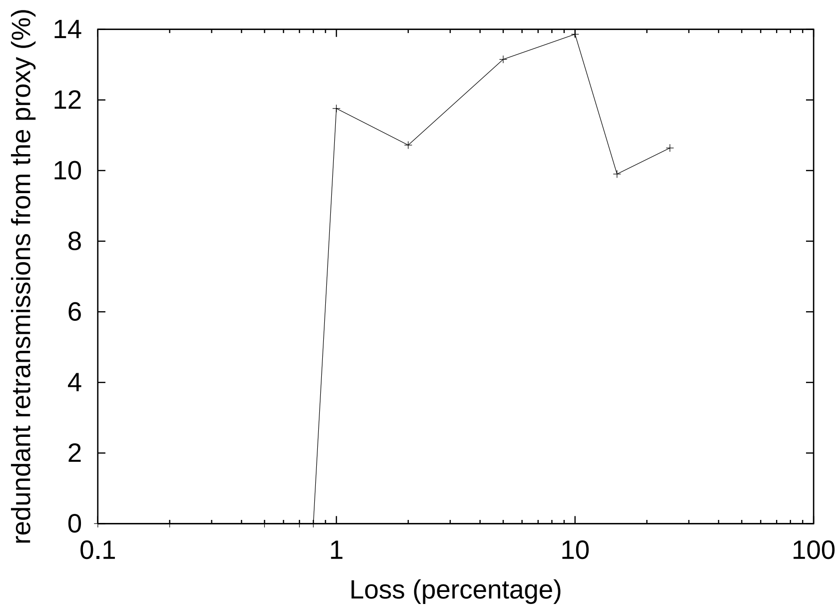

Figure 6.9: Redundant Retransmissions from the Proxy (expressed as a percentage of total retransmissions)

error rates. In Subsection 6.4.3 the effect of the proxy on the behaviour of various TCP variables is examined.

6.4.2 Behaviour of the TCP Variables without the Proxy

As was observed in the measurement study, the increase in completion time with increasing packet loss is due not only to the additional retransmissions but also to the retransmit timer backoff. It is seen from the simulation results that the retransmit timeout value (RTO) reaches the maximum value of 64 seconds for an error rate of 0.8%. Also, the difference in throughput between 0.2% and 0.3% is so high because of the RTO changing from 2 seconds to 16 seconds.

To further understand this difference, it is necessary to examine the behaviour of the congestion window, the RTO values, and the packet sequence numbers for the duration of the connection. Figures 6.10 and 6.11 show the extent to which the RTO has backed off. Because of the backoff of the retransmit timer, a packet would be retransmitted after 16 seconds in the 0.6% loss case, whereas it would be retransmitted only after 64 seconds in the 5% case. During this time the sender is

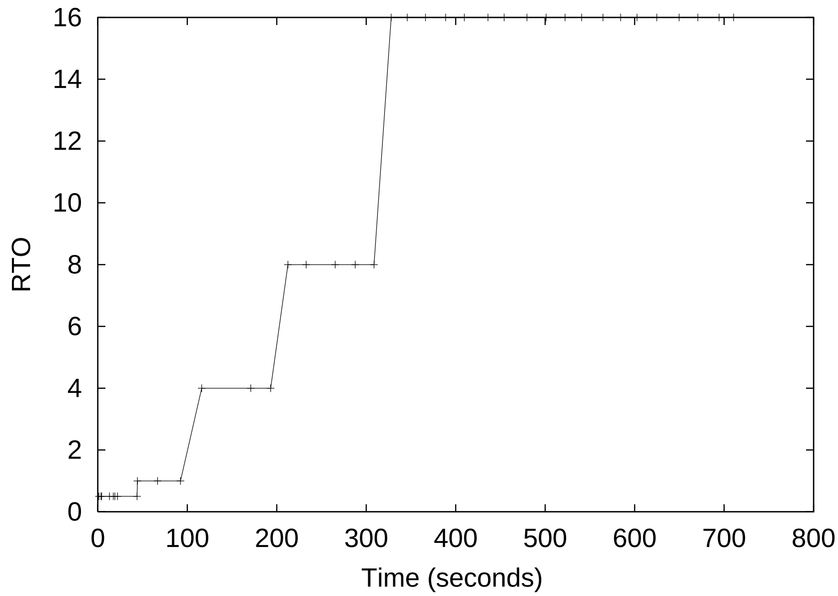

Figure 6.10: Behaviour of the Retransmit Time Out (RTO) Value for 0.5% Loss

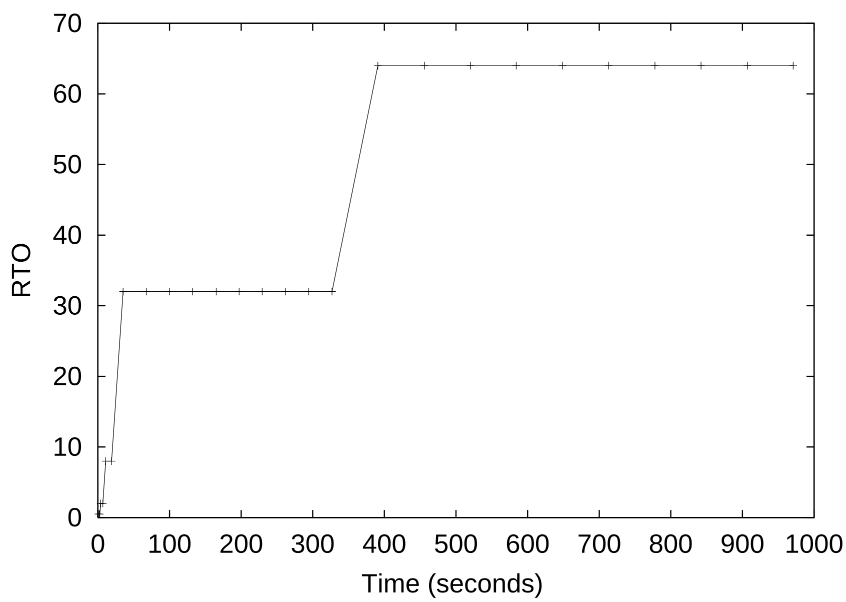

Figure 6.11: Behaviour of the Retransmit Time Out (RTO) Value for 5% Loss

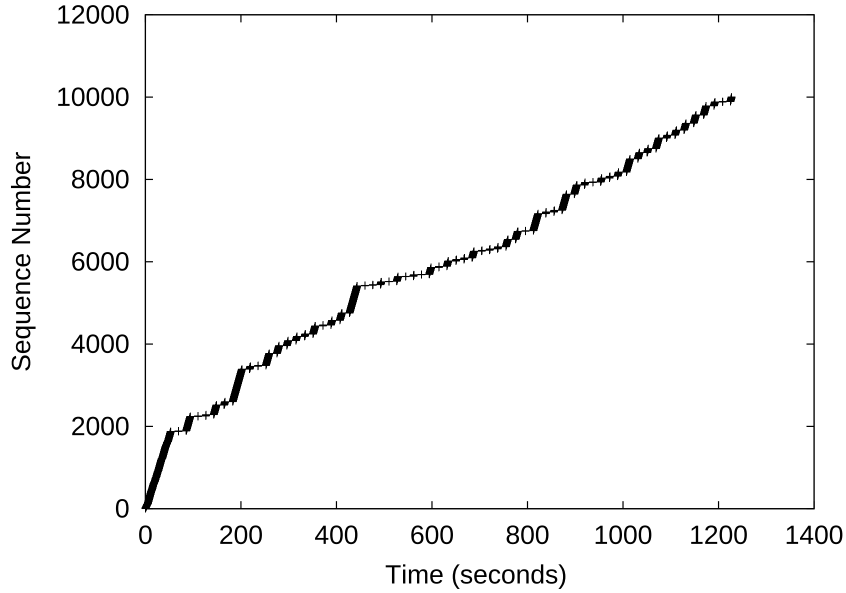

Figure 6.12: Complete Trace of Packet Sequence Numbers for 0.7% Loss

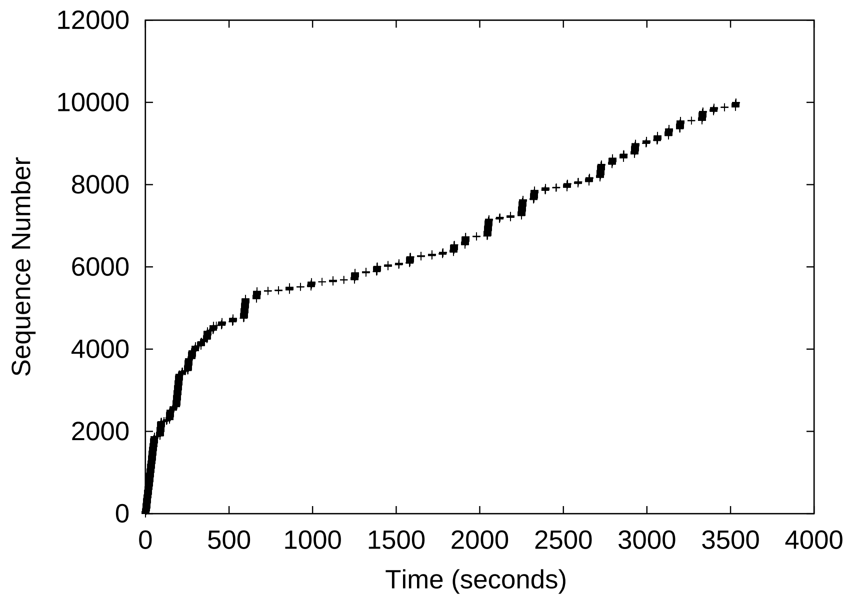

Figure 6.13: Complete Trace of Packet Sequence Numbers for 0.9% Loss

simply waiting without sending data packets into the network. This waiting period contributes to a significant increase in completion time and a decrease in throughput. This waiting period is shown more clearly in the traces of packet sequence numbers.

Figures 6.12 and 6.13 present the sequence numbers of the packets for the entire duration of the connection. To better portray the retransmissions, enlarged

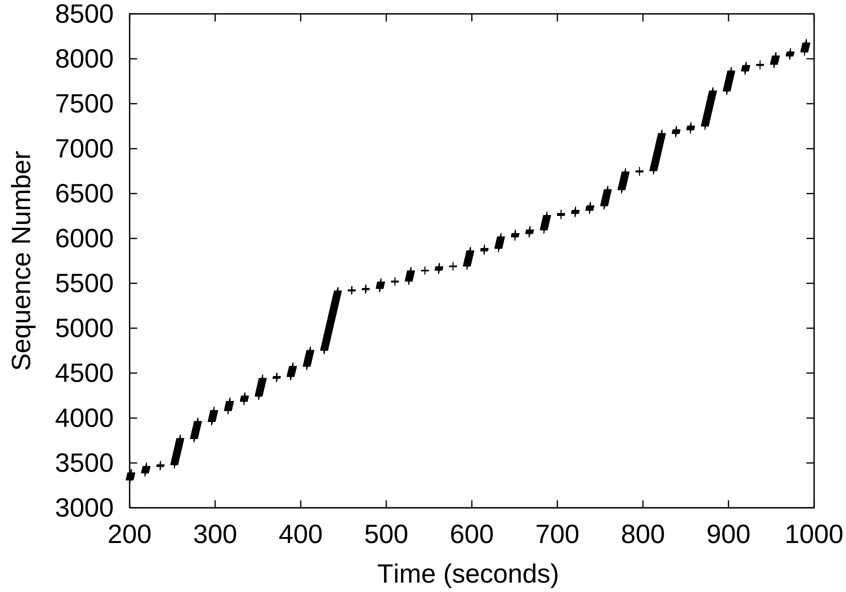

Figure 6.14: Segment of Trace of Packet Sequence Numbers for 0.8% Loss

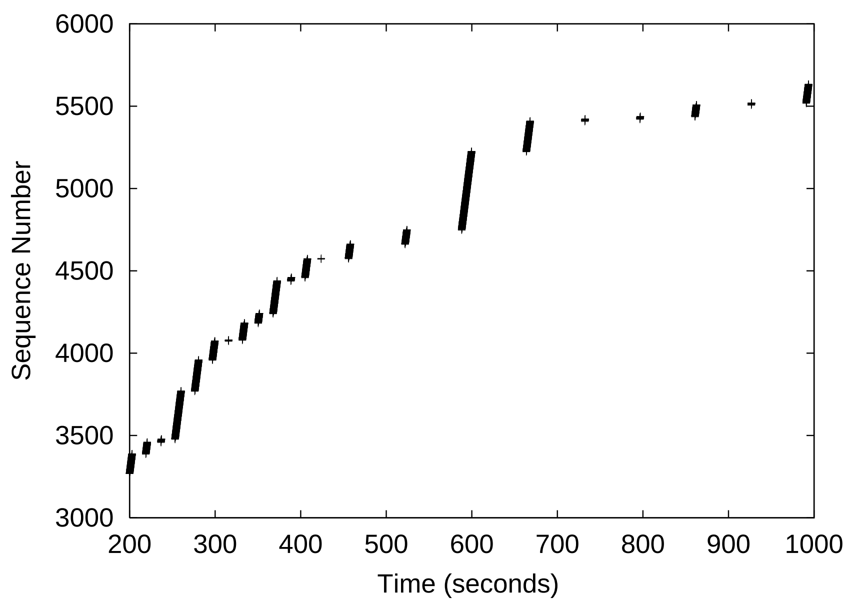

Figure 6.15: Segment of Trace of Packet Sequence Numbers for 0.9% Loss

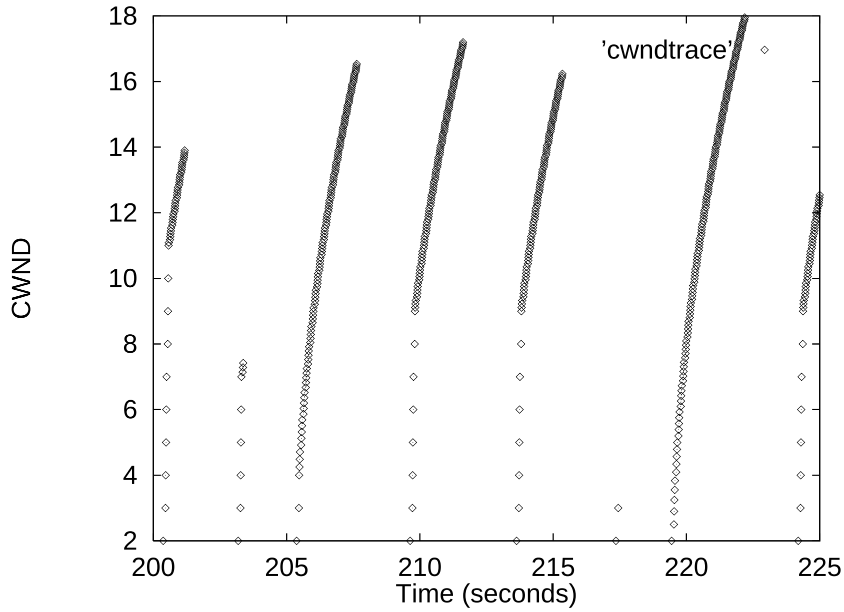

Figure 6.16: Behaviour of the Congestion Window for 0.6% Loss

versions of these traces in the region between 200 and 225 seconds are presented in Figures 6.14 and 6.15. In these figures, the time before the sender retransmits is seen to be greater for 0.9% loss than for 0.8% loss because of the backoff nature of the retransmit timer. During these periods, the sender is not sending any packets but rather is waiting for ACKs from the receiver. As mentioned earlier, these spurts of idle time at the sender contribute to a significant increase in completion time and a corresponding decrease in throughput.

Figures 6.16 and 6.17 show the behaviour of the congestion window with error rates of 0.6% and 0.7% respectively. Since TCP is unable to distinguish between loss on the wireless link and congestion, the congestion control algorithms are invoked by the TCP sender. The sender retracts to the slow start phase after a timeout. With increased loss, due to the increase in the number of timeouts, the sender is in the slow start phase for a greater amount of time. This means that the sender is transmitting data at a slower rate.

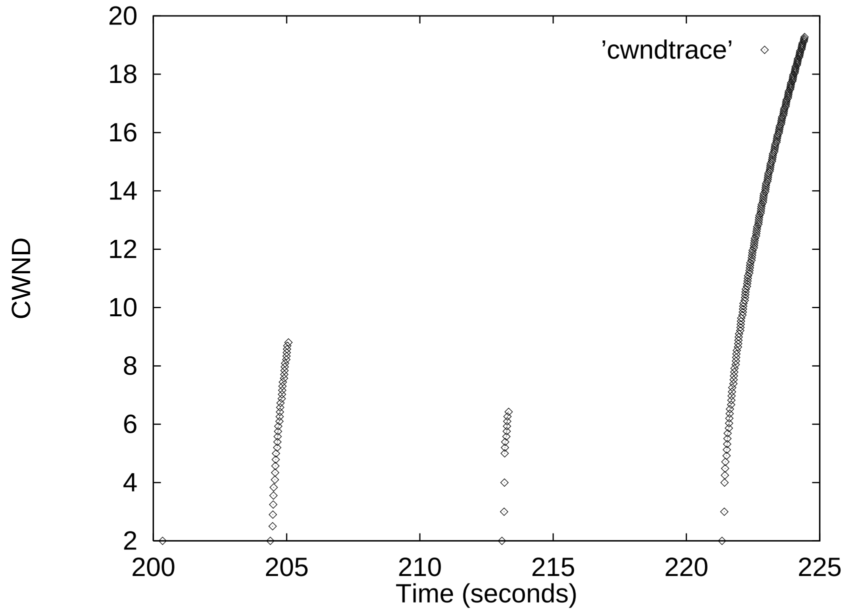

Figure 6.17: Behaviour of the Congestion Window for 0.7% Loss

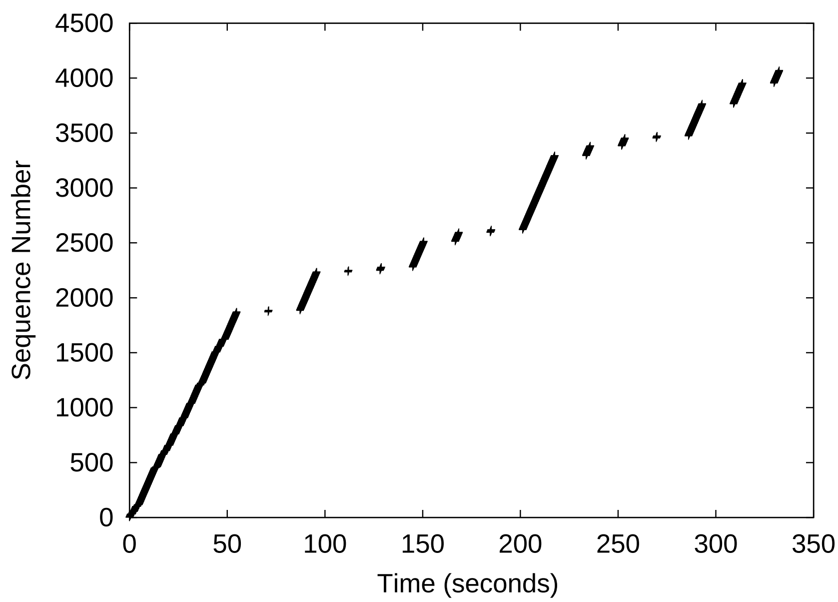

Figure 6.18: Packet Trace (without Proxy)

6.4.3 Behaviour of the TCP Variables with the Proxy

In this subsection, the behaviour of the TCP variables is examined in the presence of the proxy.

Figure 6.18 presents the sequence numbers of packets for a 5% packet loss without

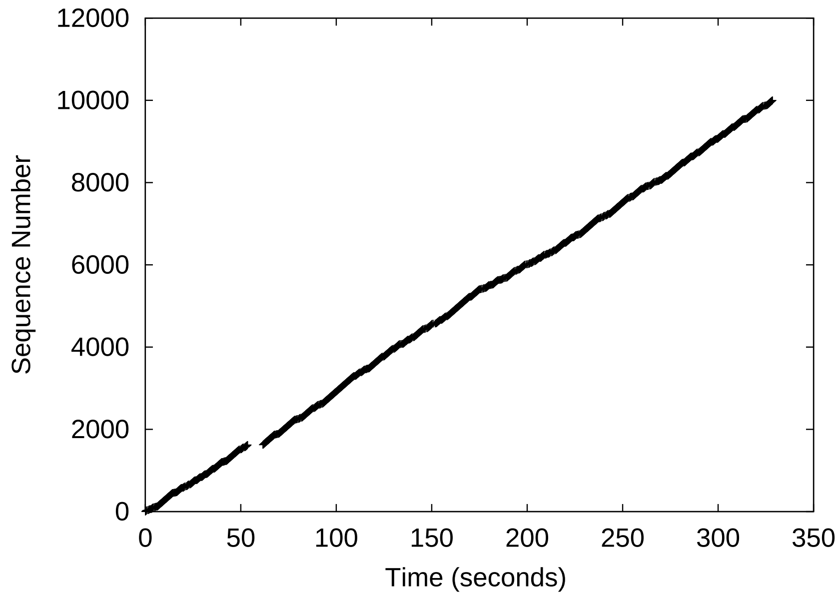

Figure 6.19: Packet Trace (with Proxy)

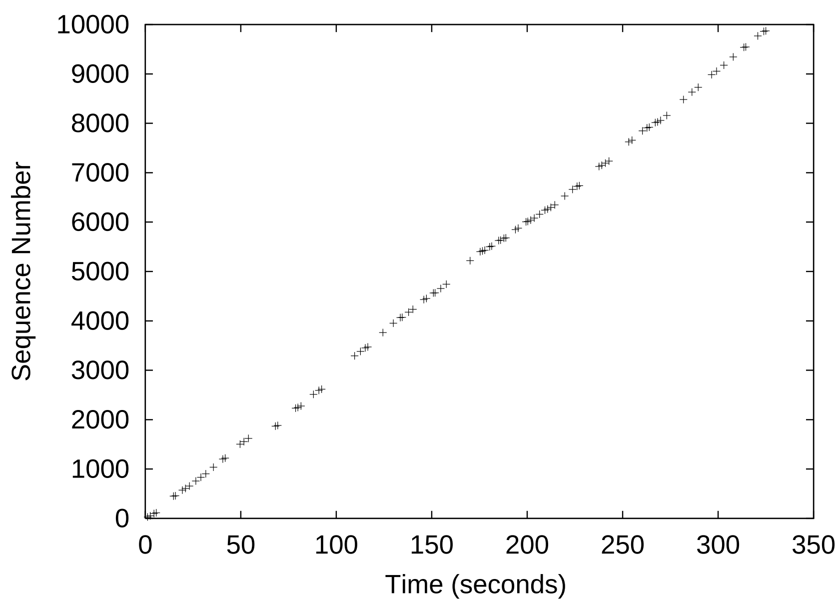

Figure 6.20: Packet Trace at the Proxy

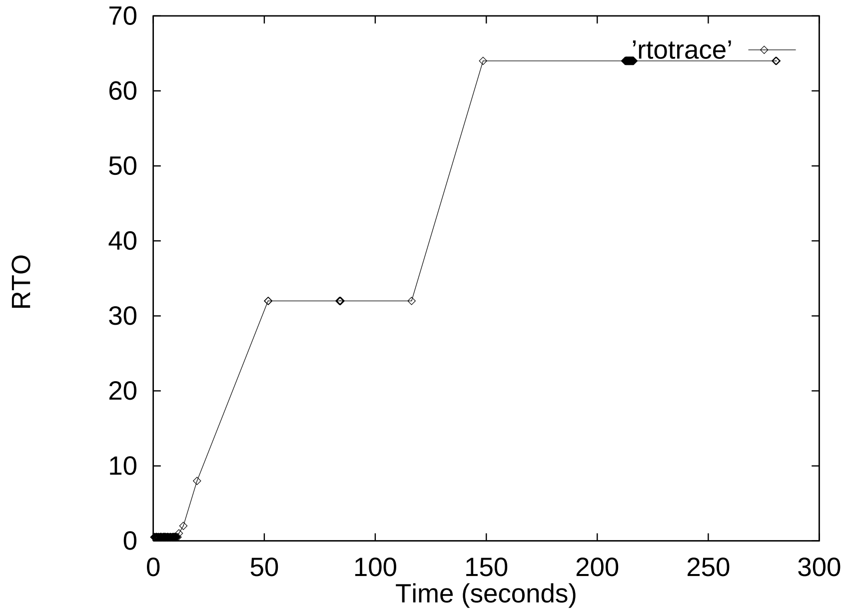

Figure 6.21: Behaviour of the Retransmit Timeout (RTO) Value for 0.6% Loss (without Proxy)

the proxy. The gaps show the times at which the sender is idle without sending any packets due to the backoff of the retransmission algorithm. Figure 6.19 shows the sequence numbers at the sender with proxy but unlike in Figure 6.18, packets are being transmitted even during those time intervals. It can be seen that more packets are transmitted in less time, resulting in reduced completion times which in turn means greater throughput. This is because of the local retransmissions from the proxy as shown in Figure 6.20. These local retransmissions generate ACKs from the receiver that travel back to the sender, thereby preventing timeouts at the sender.

Figures 6.21 and 6.22 show the behaviour of the RTO values with the proxy and without the proxy. In Figure 6.21, for example, a packet that is lost at t = 200 seconds will be retransmitted after 64 seconds, whereas, with the proxy, the packet that is lost at t = 200 seconds will be retransmitted after only 4 seconds.

Figures 6.23 and 6.24 show the behaviour of the congestion window with the proxy and without the proxy. The congestion window gives an indication of the rate at which the sender can transmit data into the network. Whenever a timeout occurs, the congestion window is re-initialized to 1 and hence the rate decreases.

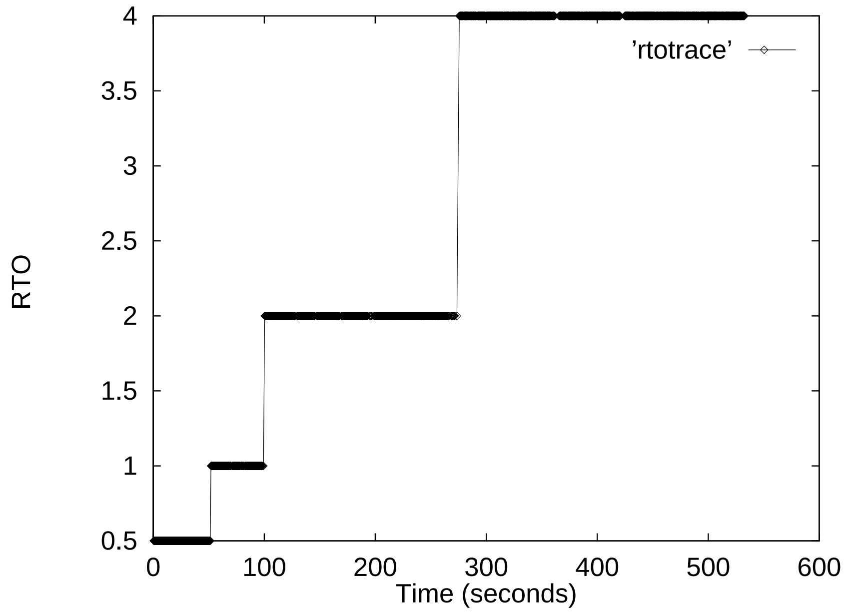

Figure 6.22: Behaviour of the Retransmit Timeout (RTO) Value for 0.6% Loss (with Proxy)

The proxy not only reduces the number of retransmissions from the sender, but also reduces the extent to which the timer backs off at the sender. When the sender starts retransmitting, the rate at which it does so is higher in the presence of a proxy as shown in Figure 6.24. Effectively in the presence of a proxy, the sender is transmitting data at this higher rate for a longer amount of time. This also contributes to the increased throughput in the presence of a proxy.

6.5 Experiments using the Deterministic and Bursty Loss Models

With deterministic loss, there is no multiple packet loss unless the loss is of the order of the window size. So if the window size is large even a smaller loss will have a greater impact because of the backoff of the timeout mechanism.

While there is a significant difference in the nature of errors exhibited by the deterministic and bursty loss models, there is still a significant improvement in the end-to-end TCP throughput while using a proxy as shown in Figures 6.27 and 6.28.

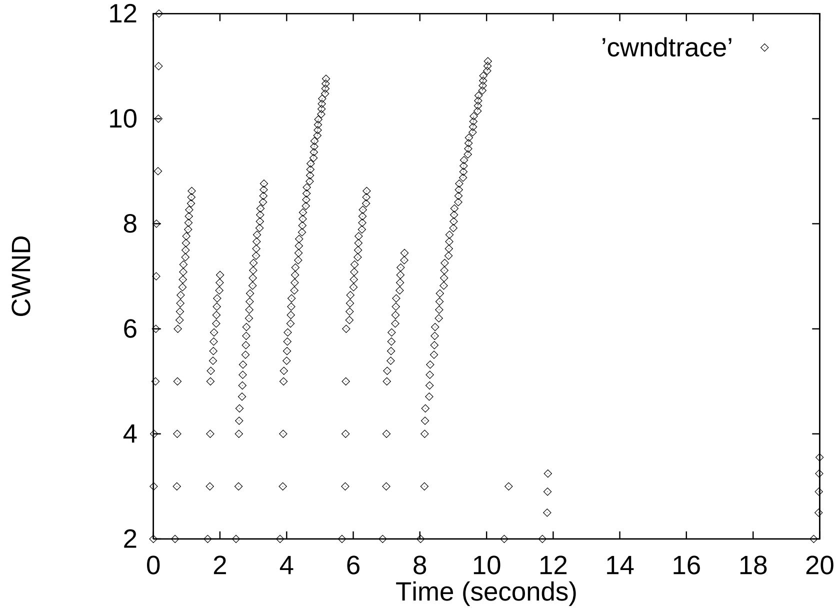

Figure 6.23: Behaviour of the Congestion Window for 5 % Loss (without Proxy)

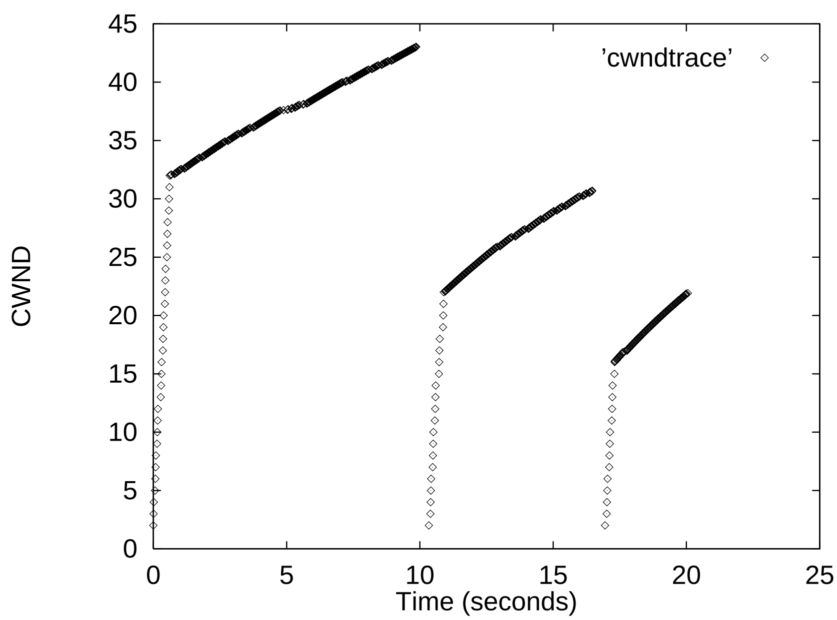

Figure 6.24: Behaviour of the Congestion Window for 5% Loss (with Proxy)

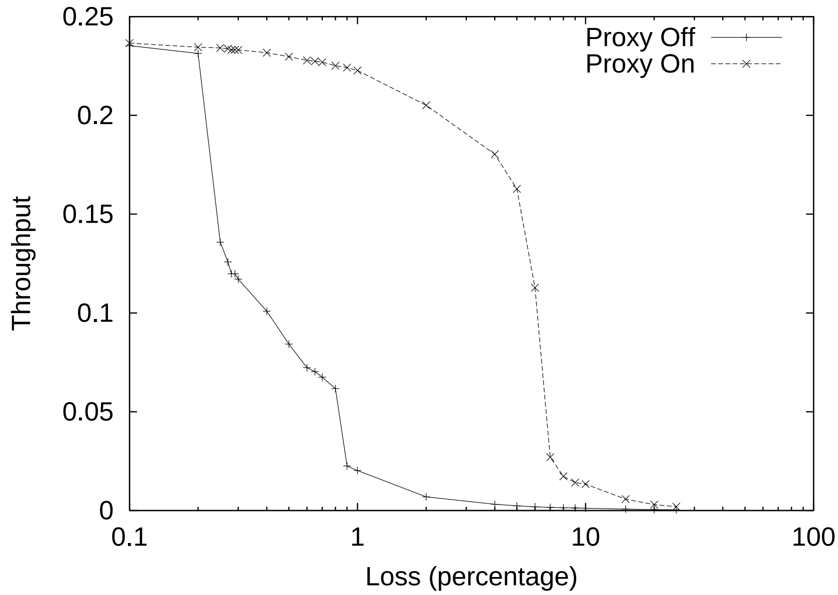

Figure 6.25: Comparison of Throughput with and without Proxy with the Deterministic Error Model

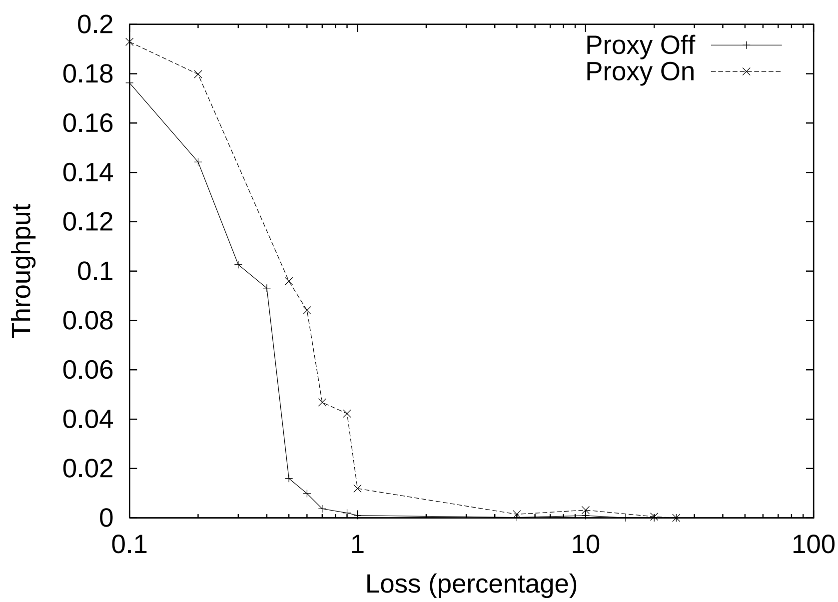

Figure 6.26: Comparison of Throughput with and without Proxy with the Bursty Error Model

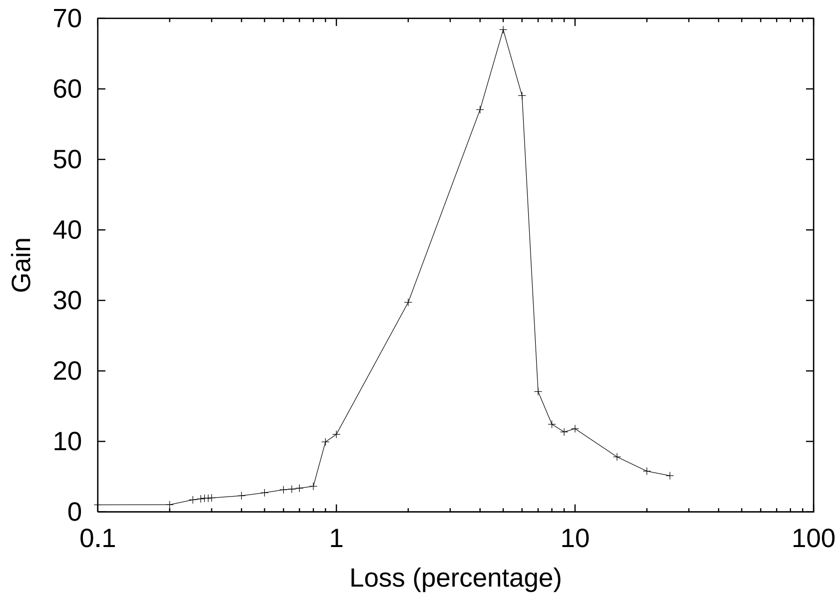

Figure 6.27: Gain in Throughput with the Proxy and the Deterministic Error Model

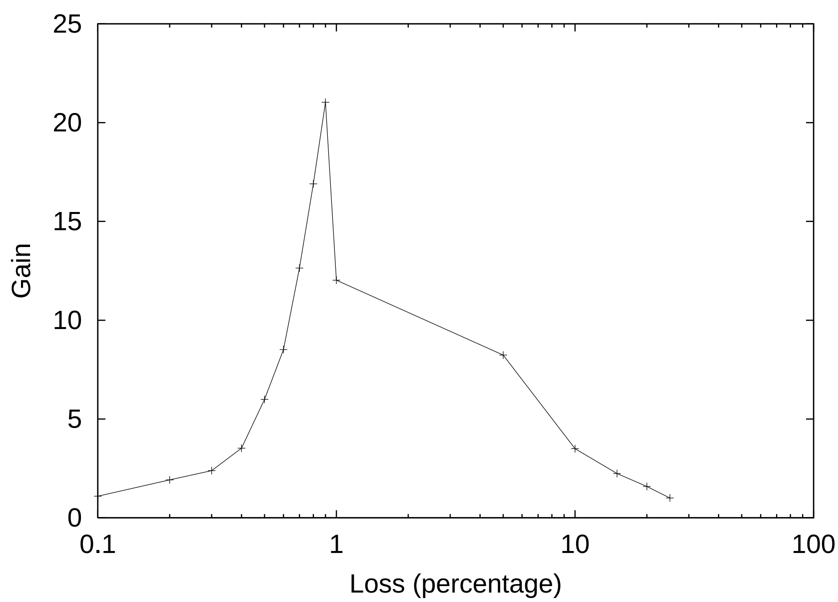

Figure 6.28: Gain in Throughput with the Proxy and the Bursty Error Model

The extent of this improvements might be effected by the nature of the losses though. The maximum gain obtained in the presence of a proxy is significantly lower for the bursty error model as shown in Figure 6.28. This is because bursty losses disrupt the self-clocking model (as described in A.1) to a greater extent than the other types of losses. So, it is more difficult for the TCP connection to achieve steady state in the presence of bursty losses.

6.6 Conclusions

The introduction of a proxy mechanism decreases the number of retransmissions from the sender, thereby contributing to a significant increase in the throughput. The extent to which the RTO backs off at the sender is also affected by the presence of a proxy. The proxy mechanism also affects the rate of data transfer from the sender. In this way the proxy mechanism tries to address all the problems mentioned in the measurement study. The proxy mechanism will be efficient only if the ACKs reach the original sender before the RTO timer expires at the sender. If the retransmission from the proxy arrives at the receiver but still does not prevent a timeout at the sender, this causes a redundancy in functionality. It was also seen that at high error rates, timeouts at the sender cannot be completely eliminated even in the presence of a proxy. However timeouts are reduced in number, contributing to increased throughput. As observed in the results, the gain in throughput with the proxy is affected by both the number of timeouts at the sender and the maximum RTO value reached. The proxy provides the highest gain in throughput when both these variables are affected. The proxy thus has a smoothing effect on the losses on the wireless links.

6.7 Summary

In this chapter, the use of a proxy as a possible solution to the problems encountered by TCP's response in the presence of wireless links was examined. Additional metrics are proposed to help in this evaluation. As per the design described in the previous chapter, the proxy attempts to solve the problems locally, and tries to shield the sender as much as possible from the effects of the wireless link. The use of three different types of loss models provides a better justification for the overall results than a single model. The simulation model that was developed for these experiments can be tested for various designs. The investigator's primary interest might be to modify the system model parameters in order to find the design that maximizes one or more performance measures or simply to study the behaviour of the system.

Chapter 7

Summary and Conclusions

7.1 Summary

Since many Internet applications use TCP as the transport layer for reliable data transfer, the performance of TCP directly affects the performance of these applications. For wireless connections to integrate seamlessly into the Internet, TCP performance in a wireless environment is of significant importance. Some of the assumptions made during the design of TCP are no longer valid in a wireless environment. This thesis investigated the various protocol issues that would arise at the transport layer as wireless technologies are integrated into the existing network infrastructure. The thesis also explored the use of a proxy mechanism as a possible solution to the problems experienced by TCP in a wireless environment.

There were several goals for this research: to evaluate the performance of TCP in an experimental wireless testbed, and to evaluate the use of a proxy as a possible solution to alleviate the problems experienced by TCP in a wireless environment through simulation. This work consisted of two distinct phases: a measurement study of TCP performance, and a simulation study of a proxy mechanism. The measurement study gave a detailed analysis of how the lossy wireless links impacted the various TCP mechanisms. Losses on the wireless links led to severe degradation in the end-to-end throughput of a TCP connection. The measurement study demonstrated how TCP cannot distinguish between packet loss due to congestion and packet loss due to errors on the wireless link. TCP assumed the loss on the wireless link to be congestion and invoked the congestion control mechanisms. This resulted in increased completion time for the data transfer and thereby lower throughput. In the simulation study, a model for a proxy mechanism was developed to address these problems. The proxy mechanism shields the sender from losses on the wireless link by performing local retransmissions. The use of a proxy does not require any modifications to the TCP source code at the sender and hence is transparent to the sender. The performance of the proxy mechanism was evaluated using simulations and it was found that the proxy mechanism significantly improves the performance of a TCP connection for low and moderate error rates. For higher loss rates on the wireless link, the proxy mechanism could not completely shield the sender from the affects of loss on the wireless link.

7.1.1 Contributions

This work is immediately applicable to the growing demand for wireless connectivity to the Internet. This work provides measurement data of TCP performance on a real wireless testbed. This work also provides a detailed analysis of TCP internals in a lossy wireless environment. Similar analysis can be performed while evaluating TCP performance in other networking environments.

A network simulator consisting of wired and wireless links was developed. This network simulator can be used to examine various network topologies and can easily be extended, for example, to accommodate models for other protocols. Three different loss models were incorporated into the simulator that can be used to model losses on the wireless link. Various TCP mechanisms were incorporated into the TCP model. The TCP models can be used for other research in TCP, congestion control mechanisms and retransmission mechanisms. Metrics were proposed for evaluating transport layer performance.

Finally this thesis also presented a framework for the design of a proxy mechanism and developed a simulation model for such a proxy mechanism. Additional metrics were also proposed for evaluating proxy performance.

7.2 Conclusions

It has been observed that while the current applications can run over wireless links, there can be a significant degradation in performance. This happens because TCP is unable to distinguish between packet loss resulting from errors on the wireless link and packet loss due to congestion on the network. So, TCP invokes the congestion control mechanism even if the packet loss is due to the lossy nature of the wireless link. This results in degraded end-to-end TCP performance, largely because of the backoff of the retransmit timeout and the reduction in the window size. When this happens the TCP sender waits a longer time to retransmit the lost packets and also slows down the rate of transmission of data thereby causing further increase in the completion times of data transfer leading to a decrease in end-to-end throughput.

The proxy mechanism provides a viable solution to the problems experienced by TCP in a wireless environment. The proxy effectively shields the sender from packet loss on the wireless link by performing local retransmissions on the wireless link. This reduces the number of timeouts at the sender, thereby reducing the extent of backoff of the retransmit timer and maintaining a higher rate of data transfer. The idle time of the sender is reduced and the completion time of the data transfer is lower. This leads to an increase in throughput. The proxy performs well for different error types although to a different degree. At higher error rates timeouts cannot be completely avoided at the sender even in the presence of a proxy. Various factors affect the effective design of a proxy, including the redundant retransmissions from the sender and the local retransmission mechanism at the proxy. Additional mechanisms are needed to obtain high end-to-end performance even at high error rates.

To summarize, the proxy mechanism provides a framework to address problems caused by wireless links by performing local retransmissions on the wireless link. The proxy mechanism also preserves protocol semantics and is transparent to the application layer. The proxy mechanism can be used to leverage the locality of packet loss.

7.3 Future Research Directions

This thesis considered a LAN environment for the experiments but the various models developed can easily be extended for simulations in a wide area network. The round-trip times in a wide area network have higher variance than observed in this study and would make it more difficult for TCP to attain the steady state after a timeout. It is expected that the proxy would perform well even in this wide area networking environment. In such an environment, greater care should be taken in the design of the local retransmission algorithm at the proxy. If the ACKs from the receiver do not reach the sender before a timeout, the chances of redundant retransmissions from the sender are higher. This work can also be extended to study the effects of both congestion and wireless losses in a wide area networking environment.

Proxies are likely to evolve into more intelligent devices and, as the network architecture evolves, they could play a major role in providing wireless access to network services. A proxy can also perform certain optimizations to data communication specific to the wireless link. For example, data can be compressed by a proxy for transfer on the wireless link, or images can be suppressed during web access. The disparity in bandwidth between wired and wireless links will continue to exist even in the future and various mechanisms can be designed at the proxy to utilize the wireless link bandwidth more efficiently. With the evolution of real-time applications that require quality of service (QOS) guarantees on the wired networks, it will soon be necessary to support QOS for mobile hosts in the future. Trying to provide QOS guarantees during handoffs while managing the completion times due to the loss rates on the wireless links is a challenge that needs to be addressed. Such mechanisms can be implemented along with the proxy mechanism on the base station. Wireless networks also need to scale to the increased network traffic due to the increase in the number of users. The proxy mechanism should also handle the increased state information due to the increased load in such networks.

The link layer problems due to wireless links can also be addressed by performing link level retransmissions. In a heterogeneous environment consisting of both wired and wireless links, the interaction between the link level retransmission mechanism and the TCP retransmission retransmission mechanism needs to be studied. Link layer protocols attempt to hide losses from the TCP sender by using local retransmissions in response to automatic repeat request (ARQ) and perhaps forward error correction (FEC) over the wireless link. While some of the problems can be solved at the link layer, it should be kept in mind that the retransmission mechanism at the link layer will be tightly coupled to the retransmission mechanism at the TCP layer. It would be interesting to see how the various mechanisms and protocols at the link layer affect the transport layer performance.

There is an increasing need to improve the quality of existing TCP/IP implementations to offer stability in the Internet. The IETF Transport Area consists of the various working groups geared towards research on TCP. There is ongoing work to enhance the TCP retransmission mechanisms and congestion control mechanisms to enhance TCP performance. While these help to better recover from congestion they do not necessarily solve the problems due to the wireless link. The proposals providing differentiated services also need to keep in mind the performance limitations associated with TCP. Protocol enhancements are required at the transport layer to support realtime applications. There are also performance problems in TCP due to asymmetry in networks in which the transmission rate in one direction is significantly greater than that in the other direction. These are especially relevant in the case of satellite links.

7.3.1 TCP over Satellite Links

Satellites are being used to extend the global Internet geographically, and will be more commonplace in the coming years. The integration of satellite links into the Internet raises the question of whether the common Internet applications can perform satisfactorily in this environment. Satellite links have certain characteristics that can affect the performance of TCP connections and hence the applications built on top of TCP. Due to the high latency of these links the performance of interactive applications might be horrendous. From the perspective of transport protocols, there are many similarities between the mobile and wireless communications environment and the satellite communications environment. Both environments experience loss caused by data corruption and link outage (non telco-grade quality links), in addition to congestion-related losses.

Although there are similarities, the design of a transport layer protocol is much more challenging in the satellite communications environment due to the high latencies of the links, severe limitations on the channel data rates and the asymmetric nature of the links. Asymmetry is also inherent in several wide-area wireless networks, where it is often the case that a base station transmits at high power to the mobile hosts, but, to reduce power consumption, the mobile hosts transmit to the base station at relatively low power. In addition these mobile hosts often have to contend with other mobile hosts to gain access to the channel. In the satellite communications environment there is also asymmetry in bandwidth, latency and packet error rate. Often the link bandwidth from the satellite to the ground station is substantially larger than the link bandwidth from the ground station to the satellite. This asymmetry is a result of various engineering tradeoffs (such as power constraints) because, for most of the applications, most of data transfer is from the satellite to the ground.

With asymmetric bandwidth, ACKs flowing over the slow reverse channel can reduce the rate of flow of data packets and disrupt TCP's self-clocking mechanism as described in Appendix A.1. One way of alleviating this problem is to reduce the size of the ACKs using SLIP header compression. Another approach could be to send the ACKs less frequently, but this might make the traffic from the sender very bursty. This is very much an open issue and needs further investigation.

The high bandwidth-delay product A of these connections has certain implications for TCP performance that need to be carefully addressed. Because of the high bandwidth-delay product, slow start and congestion avoidance algorithms behave much differently when the path includes a satellite link than for exclusively terrestrial ones. Some of the TCP extensions that have been proposed for getting better throughput over high delay-bandwidth product networks like ATM could perhaps be used for satellite links. To maximize bandwidth utilization in ATM networks, TCP needs to use a much larger window size. Large windows are needed to fill the channel capacity quickly.

It remains to be seen whether this idea can be extended to the satellite networks. For example, on a satellite link with a round-trip delay of 0.8 seconds and bandwidth of 1.54 Mbps (T1 link), the theoretical optimal window size is 154K bytes, or considerably more than a maximum window size of 64K commonly used in current implementations. In order to maximize bandwidth utilization, TCP needs a much larger window size. RFC 1323 specifies a large window option that can be used to increase the window size from 2^{16} to 2^{32}. To obtain better performance over satellite links, both sender and receiver may need to use a version of TCP that implements this extension. However, unlike in ATM networks, the impact of loss is much more pronounced in satellite networks. The impact of loss while using large windows is an issue that requires further research.

The time required for slow start to achieve full bandwidth is a function of the round-trip time and since the round-trip time over satellite links is large, the throughput for the connections using satellite links will be severely limited during slow start. The time spent in slow start will have greater negative impact on file transfers of relatively small sizes (file transfers that complete data transfer before the TCP connection attains equilibrium).

Another problem could result from congestion avoidance. When a TCP connection is sending the full bandwidth of a T1 link and a round-trip time of 640 msec, for example, detects a lost segment, it slows down the sending rate from approximately 240 segments per second to approximately 120 segments per second (assuming fast retransmit and fast recovery). During the congestion avoidance phase, TCP requires 120 round-trip times which is about one and a half minutes to return to peak rate of 240 segments. Thus in a satellite network with longer latency, bandwidth adaptation takes a longer time.

Loss recovery becomes even more critical with satellite links. In addition to the problems experienced by TCP due to loss, the large round-trip time in a satellite environment causes TCP to reach equilibrium only after a very long time, thus contributing to a further degradation in throughput. Clearly the use of maximum RTO as 64 seconds need to be increased in the satellite environment. Given the limitations of the satellite environment, the use of selective acknowledgments can also be very helpful. Unlike the cumulative ACK scheme, selective ACKs explicitly inform the sender about segments that have arrived and segments that may have been lost, thus giving the sender additional information about segments that need to be retransmitted.

Appendix A

Bandwidth-delay Product

TCP's performance does not depend on the transfer rate alone but rather on the product of the transfer rate and the round-trip time. The bandwidth-delay product is a measure of the capacity of a link and is defined as the product of the bandwidth and the round-trip time [8]. The bandwidth-delay product also corresponds to the amount of buffering required (window size) at the ends of the "pipe" for maximum throughput of a TCP connection over the path. Since the ratio of window size to round-trip delay equals the bottleneck bandwidth, TCP's congestion window size tends towards the connection's bandwidth-delay product. This is the number of outstanding bytes in transit in the steady state. This implies that a lower-bandwidth connection implies small window sizes.

The maximum window size allowed by the TCP header's 16 bit window field is $2^{16} = 65,535$ (64K) bytes. A bandwidth-delay product greater than 65,535 bytes is said to be "high." RFC 1323 [8] refers to an Internet path operating in this region as a "long fat pipe" and a network containing this path as an LFN (long fat network). To maximize the bandwidth utilization of such networks, TCP should use much larger window sizes. RFC 1323 specifies a window scaling TCP option which allows a maximum window size of up to $2^{32} = 1,073,725,440$ bytes.

For a T1 connection of 1.544 Mbits/sec, a high bandwidth-delay product would occur with a round-trip delay greater than around 340 msec. For a T1 geosynchronous satellite connection with a delay of approximately 500 msec, the bandwidth-delay product is 772,000 bits which is above 65,535 bytes ("high" according to the definition above) and thus falls in the LFN category. For a terrestrial T1 connection with 100 msec round-trip time the bandwidth-delay product is 19,300 bytes which is not "high."

Thus the bandwidth-delay product of a network path determines its capacity to hold data packets. Reliable transport protocols such as TCP attempt to match the amount of outstanding data at the sender to this capacity. Routers along the path are provisioned with buffers to accommodate temporary fluctuations in the flow of data, so the sizes of these buffers should also be added to determine the capacity of the path.

A.1 Self-Clocking

TCP uses acknowledgements (ACKs) from the receiver to implement reliable data delivery. This builds into the protocol a fundamental property called *self-clocking.* When the ACKs of previous segments reach the sender, they trigger the sending of new packets into the network. Since the ACKs cannot be generated faster than data packets get through the network, the protocol is self-clocking. This clocking makes TCP a tightly-coupled feedback system and frees the sender from maintaining explicit software timers for transmission. However, this also leads to significant performance degradation when ACK feedback is imperfect or variable.

A TCP connection is said to be in *equilibrium* if it is running (stable) with a full window of data in transit. When the self-clocking mechanism is disrupted for any of a variety of reasons, such as congestion and packet loss, it is damaging to TCP's performance because it takes additional round-trip times to reach this equilibrium state. High variability of round-trip time is detrimental to the self-clocking property. During start-up the challenge for TCP is to reach the equilibrium state quickly, without overshooting, with very little information available about the network. When a TCP connection is in the steady state, the sender transmits a window's worth of data every roundtrip at a rate equal to the bottleneck bandwidth.

References

[1] E. Ayanoglu, S. Paul, T. E. LaPorta, K. K. Sabnani, and R. D. Gitlin. AIRMAIL: A link-layer protocol for wireless networks. *Wireless Networks*, 1(1):47–60, February 1995.

[2] R. Bagrodia, W. W. Chu, L. Kleinrock, and G. Popek. Vision, issues and architecture for nomadic computing. *IEEE Communications Magazine*, 2(6):14–27, December 1995.

[3] A. Bakre and B. R. Badrinath. I-TCP: Indirect TCP for mobile hosts. In *Proceedings of 15th International Conference on Distributed Computing Systems (ICDCS)*, pages 136–143, Vancouver, BC, May 1995.

[4] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A comparison of mechansims for improving TCP performance over wireless links. In *Proceedings of ACM SIGCOMM '96*, pages 256–269, Stanford, California, August 1996.

[5] H. Balakrishnan, S. Seshan, E. Amir, and R. H. Katz. Improving TCP/IP performance over wireless networks. In *Procceedings of Mobicom95, ACM International Conference on Mobile Computing and Networking*, pages 2–12, Berkeley, California, November 1995.

[6] D. F. Bantz and F. J. Bauchot. Wireless LAN design alternatives. *IEEE Network*, pages 43–53, March/April 1994.

[7] P. Bhagwat. *A Framework for Integrating Mobile Hosts within the Internet*. PhD thesis, Dept. of Computer Science, University of Maryland, 1995.

[8] D. Borman, R. Braden, and V. Jacobson. TCP extensions for high performance. Technical Report RFC 1323, Internet Engineering Task Force, May 1992.

[9] R. Braden. Requirements for Internet Hosts – Communication Layers. Technical Report RFC 1122, Internet Engineering Task Force, October 1989.

[10] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson. TCP Vegas: New techniques for congestion detection and avoidance. In *Proceedings of ACM SIGCOMM'94*, pages 24–35, London, UK, August 1994.

[11] R. Caceres and P. B. Danzig. Characteristics of wide-area TCP/IP conversations. In *Proceedings of ACM SIGCOMM '91*, pages 101–112, Zurich, Switzerland, September 1991.

[12] R. Caceres and L. Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE Journal on Selected Areas in Communications*, 13(5):850–857, June 1995.

[13] J. C. Chang, R. A. Miska, and R. Anthony. Wireless systems and technologies: An overview. *AT&T Technical Journal*, 72(4):11–18, July/August 1993.

[14] V. Chikarmane. Network Support for Mobile Hosts in a TCP/IP Internetwork. Master's thesis, Dept. of Computer Science, University of Saskatchewan, 1995.

[15] V. Chikarmane, Carey L. Williamson, and Richard B. Bunt. Implementing Mobile IP Routing in 4.3 BSD Unix: A Case Study. *Internetworking: Research and Experience*, 6(4):209–227, 1995.

[16] D. E. Comer. *Internetworking with TCP/IP: Principles, Protocols and Architecture.* Prentice Hall, 1991.

[17] A. DeSimone, M. C. Chuah, and O. C. Yue. Throughput performance of transport-layer protocols over wireless LANs. In *Proceedings of Globecom*, pages 542–549, Houston, Texas, November 1993.

[18] R. Dube, C. D. Rais, and S. K. Tripathi. Improving nfs performance over wireless links. Technical Report CS-TR-3583, Dept. of Computer Science, University of Maryland, 1995.

[19] D. Duchamp. Issues in wireless mobile computing. In *Procceedings of Third Workshop on Workstation Operating Systems*, pages 1–7, Key Biscayne, Florida, April 1992.

[20] A. Hills and D. B. Johnson. A Wireless Data Network Infrastructure at Carnegie Mellon University. *IEEE Personal Communications*, 3(1):56–63, February 1996.

[21] T. Imienlinski and B. R. Badrinath. Wireless computing: Challenges in data management. *Communications of the ACM*, 37(10):19–28, October 1994.

[22] J. Ioannidis. *Protocols for Mobile Internetworking.* PhD thesis, Dept. of Computer Science, Columbia University, 1993.

[23] V. Jacobson. Congestion avoidance and control. In *Proceedings of ACM SIGCOMM '88*, pages 314–329, Stanford, California, August 1988.

[24] V. Jacobson and R. Braden. TCP Extensions for Long-delay Paths. Technical Report RFC 1072, Internet Engineering Task Force, October 1988.

[25] R. Jain. *The Art of Computer Systems Performance Analysis.* Wiley, 1991.

[26] V. Josyula, R. B. Bunt, and J. Harms. Measurements of TCP performance over wireless connections. In *Proceedings of Wireless 96*, pages 289–299, Calgary, Alberta, July 1996.

[27] P. Karn and C. Partridge. Improving round-trip time estimates in reliable transport protocols. In *Proceedings of ACM SIGCOMM '87*, pages 2–7, Stowe, Vermont, August 1987.

[28] L. Kleinrock. Nomadic computing – an opportunity. *ACM SIGCOMM Computer Communication Review*, 25(1):36–40, January 1995.

[29] H. Koch, L. Krombholtz, and O. Theel. A Brief Introduction into the World of Mobile Computing. Technical report, Dept. of Computer Science, University of Darmstadt, 1993.

[30] R. Kohno, R. Meidan, and L. B. Milstein. Spread spectrum access methods for wireless communications. *IEEE Communications Magazine*, 33(1):58–67, January 1995.

[31] M. Kojo, K. Raatikainen, and T. Alanko. Connecting Mobile Workstations to the Internet over a Digital Cellular Telephone Network. Technical Report C-1994-39, Dept. of Computer Science, University of Helsinki, September 1994.

[32] D. C. Kulkarni, A. Banerji, M. R. Casey, and D. L. Cohn. Information Access in Mobile Computing Environment. Technical Report 93-11, Distributed Computing Research Lab, University of Notre Dame, October 1993.

[33] W. C. Y. Lee. Overview of cellular CDMA. *IEEE Trans. on Vehicular Technology*, 40(2):291–302, May 1991.

[34] P. Maniatis, M. Roussopoulos, E. Swierk, K. Lai, G. Appenzeller, X. Zhao, and M. Baker. The mobile people architecture. *ACM Mobile Computing and Communications Review*, 3(3):36–42, July 1999.

[35] P. Manzoni, D. Ghosal, and G. Serazzi. Impact of mobility on TCP/IP: An integrated performance study. *IEEE Journal on Selected Areas in Communications*, 13(5):858–867, June 1995.

[36] B. Marsh, F. Douglas, and R. Caceres. System Issues in Mobile Computing. Technical Report MITL-TR-50-93, Matsushita Information Technology Laboratory, Princeton, New Jersey, 1993.

[37] S. McCanne and V. Jacobson. ftp://ftp.ee.lbl.gov/tcpdump.tar.z.

[38] J. C. Mogul. Observing TCP dynamics in real networks. *ACM Computer Communication Review*, 22(3):1–7, August 1992.

[39] S. Narayanaswamy, S. Seshan, E., R. Brodersen, F. Burghardt, A. Burstein, Y. C. Chang, A. Fox, J. M. Gilbert, R. Han, R. H. Katz, A. C. Long, D. G. Messerschmit, and J. Rabaey. Application and Network Support for Infopad. *IEEE Personal Communications*, 3(2):4–17, April 1996.

[40] Bay Area Research Wireless Access Network. http://cs.berkeley.edu/randy/daedalus/barwan/barwanindex.html, 1999.

[41] G. T. Nguyen, R. H. Katz, B. Noble, and M. Satyanarayanan. A trace-based approach for modeling wireless channel behavior. In *Proceedings of the Winter Simulation Conference*, December 1996.

[42] R. Padovani and A. J. Viterbi. Implications of mobile cellular CDMA. *IEEE Communications Magazine*, 30(12):38–41, December 1992.

[43] K. Pahlavan and A. H. Levesque. Wireless data communications. *Proceedings of the IEEE*, 82(9):1398–1430, September 1994.

[44] C. Perkins. IP Mobility Support. Internet Draft, Mobile IP Working Group, IETF, January 1995.

[45] R. L. Pickholtz, L. B. Milstein, and D. L. Schilling. Spread spectrum for mobile communications. *IEEE Trans. on Vehicular Technology*, 40(2):313–322, May 1991.

[46] J. B. Postel. Transmission Control Protocol. Technical Report RFC 793, Information Sciences Institute, Marina del Rey, California, 1981.

[47] Wireless Application Protocol. http://www.wapforum.org, 1999.

[48] D. L. Schilling. Wireless communications going into the 21st century. *IEEE Transactions on Vehicular Technology*, 43(3):645–651, August 1994.

[49] B. N. Schillit, N. Adams, R. Gold, M. Tso, and R. Want. The PARCTAB mobile computing system. In *Proceedings of the Fourth Workshop on Workstation Operating Systems*, pages 34–39, Napa, California, October 1993.

[50] W. R. Stevens. *TCP/IP Illustrated*, volume 2. Addison-Wesley Publishing Company, 1995.

[51] W. Richard Stevens. *TCP/IP Illustrated*, volume 1. Addison-Wesley Publishing Company, 1994.

[52] F. Swarts and H. C. Ferreira. Markov characterization of digital fading mobile VHF channels. *IEEE Trans. on Vehicular Technology*, 43(4):977–985, November 1994.

[53] A. S. Tanenbaum. *Computer Networks.* Prentice-Hall, Inc., 1988.

[54] J. T. Taylor and J. K. Omura. Spread spectrum technology: A solution to the personal communications services frequency allocation dilemma. *IEEE Communications Magazine*, 29(2):48–51, February 1991.

[55] A. J. Viterbi. The evolution of digital wireless technology from space exploration to personal communication services. *IEEE Transactions on Vehicular Technology*, 43(3):638–643, August 1994.

[56] G. M. Voelker and B. N. Bershad. Mobisaic: An information system for a mobile wireless computing environment. In *Workshop on Mobile Computing Systems and Applications*, Santa Cruz, California, December 1994.

[57] Vector Orthogonal Frequency Division Multiplexing (VOFDM). http://www.cisco.com/warp/public/784/packet/jan00/standards.html., 2000.

[58] H. S. Wang and N. Moayeri. Finite state markov channel - a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, 44(1):163–171, February 1995.

[59] R. Yavatkar and N. Bhagwat. Improving end-to-end performance of TCP over mobile internetworks. In *Workshop on Mobile Computing Systems and Applications*, Santa Cruz, California, December 1994.

[60] J. Zahorjan and G. H. Forman. The challenges of mobile computing. *IEEE Computer*, 27(4):38–47, April 1994.

[61] L. Zhang, S. Shenker, and D. Clark. Observations on the dynamics of a congestion control algorithm: The effects of two-way traffic. In *Proceedings of ACM SIGCOMM '91*, pages 133–147, Zurich, Switzerland, September 1991.