PERFORMANCE AND SECURITY ISSUES OF TCP BULK DATA TRANSFER IN A LAST MILE WIRELESS SCENARIO: INVESTIGATIONS AND SOLUTIONS

VENKATESH S OBANAIK

NATIONAL UNIVERSITY OF SINGAPORE

2003

PERFORMANCE AND SECURITY ISSUES OF TCP BULK DATA TRANSFER IN A LAST MILE WIRELESS SCENARIO: INVESTIGATIONS AND SOLUTIONS

VENKATESH S OBANAIK
(*B.Tech Electronics and Communication Engineering)*

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE (COMPUTER SCIENCE)

DEPARTMENT OF COMPUTER SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE
2003

Acknowledgements

It is often said that *"It takes an artist to turn a piece of stone into a work of art"*. Fortunately, I had the privilege of rubbing shoulders with many such *artists* who contributed in chiselling my ideas into this thesis work. This thesis would not have been possible without the:

Invaluable suggestions and encouragement of my supervisor Dr. Lillykutty Jacob,

Constructive criticisms and support of my co-supervisor Dr. Ananda A L,

Useful interactions with the research community on various mailing lists like IPv6, Iperf, Tcpdump and Linux Users Group,

Ample infrastructure and a good research environment in CIR,

Fruitful discussions with my former colleagues Saravanan, Srijith and Michael,

Lively ambience and encouraging atmosphere at CIR due to my friends Sudharshan, Rahul, Sridhar and Aurbind,

Timely help by Sridhar with $\mathrm{L^AT_EX}$,

Soothing songs on Gold 90 FM which accompanied me on lonely nights in CIR,

And wonderful episodes of Seinfeld that kept me going till the end.

Finally, I extend my gratitude to everybody, who in one way or the other rendered their support and help.

Summary

TCP was designed nearly three decades ago with some inherent assumptions. Over the years many fixes and solutions have been proposed to make TCP cope with changing network conditions. This research work investigates some of the proposed solutions, studies their applicability and/or limitations in the last mile wireless scenario and proposes novel solutions. Two specific issues are addressed in this thesis: (a) The effect of algorithms that improve the fairness of TCP congestion avoidance on slow links and long thin networks, (b) The combined issue of performance and security in a wired-cum-wireless scenario.

The first part of the thesis demonstrates that fairness algorithms have a detrimental effect on connections traversing slow links and long thin networks. Simulations and test-bed experiments substantiate this claim. Some solutions are suggested to overcome the performance degradation.

The second part of the thesis explores the limitations of existing solutions for improving TCP performance in hybrid wired-wireless networks. The thesis proposes an integrated solution for IP security and TCP performance in hybrid wired-wireless networks, traditionally dealt with in a mutually exclusive manner. The novel scheme called the SPEP (Secure Performance Enhancing Proxy) ensures end-to-end security, enhances TCP performance, and offers multifarious benefits over the existing schemes. The SPEP scheme was implemented in FreeBSD 4.5 and performance tests were conducted in a controlled test-bed setup. The results show remarkable improvement in TCP performance in a "last mile wireless" scenario.

Contents

List of Figures

List of Tables

"The time to begin writing an article is when you have finished it to your satisfaction. By that time you begin to clearly and logically perceive what it is you really want to say."

- Mark Twain

1

Introduction

The thesis studies the existing solutions for improving the performance of TCP, investigates the applicability and/or limitations of the existing schemes in "the last mile" wireless scenario and proposes efficient solutions for it. The access network is colloquially known as "the last mile". The connection from local service provider to the consumers is referred to as the "local loop" or "the last mile" [1]. Originally developed to support telephony traffic, the "local loop" of the telecommunications network now supports both voice and Internet traffic. Different media can be used to provide "the last mile" connectivity, such as telephone wire, coaxial cable, fibre optics, satellite communications and wireless RF [1]. Wireless access to the Internet is referred to as the "last mile wireless" scenario. The "last mile wireless" is seen as a viable and cost-effective solution for last mile connectivity [2]. Hence, it becomes essential to take a second look at the existing solutions in the

context of the last mile wireless scenario. The following sections of this chapter describe the motivation for the research work, research objectives, contribution and the organization of the thesis, respectively.

1.1 Motivation

TCP was designed nearly three decades ago and fine tuned over the years for traditional networks comprising of wired networks and fixed hosts. However, the networks have changed over the years from wired to wireless, low bandwidth to very high bandwidth, stationary host to mobile host, and infrastructure based networks to ad-hoc networks. Meanwhile, Internet applications have become more demanding and versatile. Among today's applications are interactive applications demanding a quick response time, bulk data transfer applications requiring high throughput and multimedia applications sensitive to jitter. Nevertheless, TCP continues to be the most widely used transport layer protocol. In an attempt to equip TCP to make better use of the network and meet the demands of the user applications, many solutions have been proposed. The thesis investigates two specific solutions proposed to fine tune TCP and discusses the limitations of such schemes in the last mile wireless scenario and proposes solutions to circumvent the encountered problems.

1.2 Research Objectives

The thesis aims to study the existing solutions for improving the performance of TCP and to identify specific issues for further study. Investigations into the issues concerning the applicability and/or limitations of existing schemes in the context of the last mile wireless scenario and providing efficient solutions constitute the over-all objectives of the thesis.

1.3 Thesis Contribution

The thesis identifies two issues with the existing schemes designed to improve the performance of TCP: (i) The detrimental effects of fairness algorithms on the performance of slow links [3] and Long Thin Network [4](LTN); (ii) The limitations of the existing performance enhancement schemes for TCP over wireless to function in an end-to-end IPSEC environment.

Simulation and test bed experiments were conducted as part of the detailed investigations to study the effect of algorithms that improve the fairness of TCP congestion avoidance on the performance of slow links and LTN. Our results show that the fairness algorithms have adverse effects on connections traversing either slow links or LTN. We argue that it is not appropriate to apply the fairness algorithms for connections that traverse slow links or LTN. We have studied some of the possible solutions (increasing the last-hop router buffer size, reducing the advertised

window of the receiver and selectively disabling the fairness policies) in order to circumvent the adverse effects of fairness algorithms in the last mile scenario. We show that the impact can be reduced by selectively turning off the policies for slow link or LTN connections. In the second part of the thesis, we present a detailed survey of the co-existence of security and performance enhancing schemes in the last mile wireless scenario. We expose the limitations of the existing solutions in providing both end-to-end security and improved transport layer performance. We propose an innovative mechanism, which we call Secure Performance Enhancing Proxy (SPEP) to address the seemingly arduous problem of enhancing TCP performance over wireless networks, preserving end-to-end TCP semantics as well as ensuring end-to-end security. We have implemented the proposed scheme in FreeBSD 4.5 and conducted experiments in a controlled test bed setup. Our results show improved TCP performance in a secured environment with introduction of about 7 % overhead when compared to the end-to-end ELN scheme in a WAN scenario with high error rates of 1 error in every 16KB of data.

1.4 Thesis Organization

The thesis is organized as follows. Chapter 2 describes related work for enhancing the performance of TCP data transfer in a last mile wireless scenario and identifies two specific issues: (i) The effect of algorithms that improve the fairness of TCP congestion avoidance on the performance of slow links and LTN; (ii) The combined issue of performance and security in a last mile wireless scenario. Chapter 3

presents the setup for simulation and the test-bed experiments conducted to study the effect of fairness algorithms on performance of slow links and LTN and provides recommendations. Chapter 4 presents our novel approach called SPEP which provides a solution for the co-existence of IPSEC and performance enhancing solutions. Chapter 5 describes our test methodology to evaluate the performance of SPEP and presents performance results. We conclude with an indication of future work in Chapter 6.

"To look backward for a while is to refresh the eye, to restore it, and to render it more fit for its prime function of looking forward"

- Margaret Fairless Barber

2

Background Work

This chapter introduces the reader to the prior work related to improving the performance of TCP and describes the specific issues addressed by the thesis. The following sections discuss congestion control mechanisms of TCP, the issues with TCP congestion avoidance and control, and solutions proposed to address the issues.

2.1 TCP Congestion Avoidance and Control

TCP is an end-to-end connection-oriented transport layer protocol which ensures reliable transfer of data. TCP uses a window based congestion control algorithm to reduce congestion in the network. It is a self-clocking protocol and automatically

adjusts to the bandwidth of the network. At the start of the connection, TCP probes the network capacity by sending out packets at an increasingly exponential rate. This is the *slow start* phase and it continues until the slow start threshold is reached or a packet is lost. TCP then enters the *congestion avoidance* phase and sends out packets at a linear rate.

2.2 Issues with TCP Congestion Avoidance and Control

TCP congestion avoidance and control [5] was originally proposed by Van Jacobson in one of the seminal papers. It was proposed to solve a series of 'congestion collapses' that occurred during 1986. The congestion avoidance and control mechanism, later supplemented with fast recovery and fast retransmit mechanisms became the de facto standard [6] for TCP. However, there are some issues with TCP congestion control. Two specific issues are discussed in the following subsections.

2.2.1 Unfairness of TCP Congestion Avoidance

Fairness is an important criterion in the design of congestion control mechanisms. One way to define fairness is that if multiple TCP connections share a bottleneck link, the available bandwidth is shared equally among all the connections. However, it is seen that when a bottleneck link is shared by multiple connections with short and long round trip times (RTTs), the short RTT connections get a greater share of the bottleneck bandwidth [7, 8]. TCP uses the slow start mechanism to probe

the network at the start of a connection, time spent in the slow start phase is directly proportional to the RTT. And for a long RTT connection, it means that TCP stays in the slow start phase for a longer time when compared to a short RTT connection. This drastically reduces the throughput of short duration TCP connections. Furthermore, following each packet loss, TCP enters the congestion avoidance phase or even the slow start (in case of retransmission timeout). During the congestion avoidance phase, the TCP sender increases its congestion window by atmost 1 segment after each RTT [5], thus the connections with long RTT open up their congestion windows relatively slower when compared to the connections with short RTT. In an attempt to counter the bias of the congestion avoidance mechanism against long RTT connections, and in effect, to improve the fairness, many policies have been proposed [7, 9, 10] which will be discussed in Section 2.3.1. The policies were designed to enable long RTT connections to open up their congestion windows relatively fast.

2.2.2 Inability to Identify the Nature of Loss

TCP was designed for wired networks with an inherent assumption that packet loss caused by damage in the network is very small and that the loss of a packet always signals congestion [5]. TCP congestion avoidance and control procedures are invoked on detection of a packet loss. The occurrence of packet loss is indicated by either a retransmission timer timeout or the receipt of duplicate acknowledgements [6, 11]. However, packet loss can occur for reasons other than congestion. Communication over wireless links is affected by high bit error rate, temporary disconnections, high

latencies and low bandwidth. Losses due to bit-error rate and mobility of devices has a significant effect on the dynamics of TCP resulting in sub-optimal performance and reduced throughput for the connection.

2.3 Solutions Proposed to Address the Issues

In this section, we discuss the various solutions proposed to address and resolve the issues mentioned in Section 2.2.1 and Section 2.2.2.

2.3.1 Algorithms That Improve Fairness of TCP Congestion Avoidance

In an attempt to counter the bias of TCP congestion avoidance against long RTT connections, various alternate congestion avoidance policies have been proposed. The "Constant Rate" [9] algorithm was one of the proposed solutions. In this scheme it is suggested that congestion window be increased by 'c $*$ r^2 ' segments for each RTT, where 'c' is some fixed constant and 'r' is the average round trip time. In the standard congestion avoidance algorithm, the congestion window is increased at the rate of approximately 1 segment every RTT. If 'r' is the average RTT of the connection, the increase in throughput of the connection would be '1/r' segment/s every 'r' seconds. This means the rate of increase in throughput is '$1/r^2$' segments/s/s. Therefore the long RTT connections suffer. The suggestions in [8,9] was to modify the additive increase policy so that all connections, irrespective of their RTT, increase their sending rate similarly. Hence, it was referred to as

"Constant-Rate" window increase algorithm. However, the choice of a proper value for the constant 'c' is an open problem. According to the studies conducted in [7], the fairness properties were best at values of 'c' less than 100 and large values of 'c' made the connections very aggressive. However, smaller values of 'c' resulted in under utilization of the link. As mentioned in [7], in reality, the choice of a proper value for 'c' is not possible.

"Increase-by-K" (IBK) [7] was another policy that was suggested. The IBK policy was designed so that long RTT connections could increase their own throughput without co-operation from other connections. The IBK policy suggested that the congestion window should be increased by 'K' segments every RTT. The policy was to be selectively enabled, only on the long RTT connections. The values of 'K' up to 4 was recommended for good performance [7].

Another algorithm that was proposed was Congestion Avoidance with Normalized Interval of Time (CANIT) [10]. In the long RTT connections the arrival of ACK packets is relatively slow, compared to short RTT connections. CANIT addresses the fairness problem in this perspective. CANIT introduces a new parameter called Normalized Interval of Time (NIT). The congestion avoidance mechanism is modified to increase the congestion window at the rate of $\frac{RTT}{NIT} \cdot \frac{1}{Cwnd}$ on receipt of every ACK. Thus, all the connections increase their congestion window by the same amount after each interval NIT. CANIT with NIT value of 30ms was considered to be most fair.

2.3.2 Performance Enhancing Schemes for TCP over Wireless

The performance enhancing schemes can be broadly classified into three categories:

- Link-layer schemes
- Split Connection schemes
- End-to-End Schemes

A. Link-layer Schemes

The link-layer schemes address the problem from the perspective that the cause of suboptimal performance of TCP over wireless is the transmission error that occurs on the wireless link. Hence, the link-layer schemes propose reliable link-layer protocols to address the problem. The link-layer protocols employ two classes of techniques (i) error correction using forward error correction (FEC), and (ii) retransmission of lost packets using automatic repeat request (ARQ). The link-layer protocols for digital cellular techniques CDMA and TDMA primarily use ARQ. The AIRMAIL [12] protocol uses a combination of FEC and ARQ for loss recovery.

B. Split Connection Schemes

Split connection schemes attribute the performance degradation of TCP over wireless to the inability of TCP to cope with the dynamics of wireless link. The split connection schemes as the name suggests, splits the TCP connection from sender

to receiver at the base station referred to as the mobility support node. One connection is established between the sender and the base station and the other from the base station to the receiver. Split connection schemes such as I-TCP [13]use regular TCP for its connection over the wireless link, while other schemes such as MTCP [14], recommend a protocol optimized for wireless links to be used for the connection between the base station and the receiver.

C. End-to-End Schemes

End-to-end schemes abide by the principle that end-to-end argument is one of the architectural principles in the design of the Internet. Hence, all problems with TCP have to be solved end-to-end. The schemes which maintain the end-to-end semantics are Freeze-TCP [15], TCP HACK [16], SACK [17]. Freeze-TCP proposes an end-to-end solution to enable TCP to cope with long periods of disconnection due to degraded wireless link. The TCP receiver in Freeze-TCP advertises a zero window in case of an imminent link failure. The sender reacts to the zero window advertisement by freezing all retransmit timers and entering persist mode. TCP HACK [16] proposes a scheme, in which the receiver can distinguish between the nature of loss congestion or corruption. The information about the nature of loss is conveyed to the sender. The sender retransmits the packets lost due to corruption without invoking congestion control while the packets lost due to congestion are handled normally. SACK [17] provides a mechanism which enables the TCP sender to recover from multiple losses in a window of data transmitted. The SACK enabled receiver uses the TCP SACK option to acknowledge the blocks of data received in

sequence. The sender retransmits the lost segments, thereby reducing the number of probable retransmission timeouts.

D. Other Schemes

The Snoop protocol [18] proposed by Hari et al is another performance enhancement scheme, which was designed with the intent that local problems should be solved locally. Therefore, Snoop suppresses the duplicate acknowledgements that signal loss in wireless link and locally retransmit the lost segments. Thereby, Snoop achieves a remarkable improvement in the performance of TCP over wireless links.

2.4 Limitations of the Proposed Solutions

The solutions proposed in Section 2.3 overcome the shortcomings of TCP mentioned in Section 2.2. However, the solutions take a myopic view of the problem. Although the proposed solutions attempt to resolve the issue at hand, they have some limitations which restrict their applicability. In the following chapters, the drawbacks are exposed and solutions are proposed to counter the same.

2.5 Summary

This chapter presents a brief description of TCP congestion avoidance and control and the issues concerning it. Two specific issues with TCP congestion avoidance and control mechanism are identified for further examination, namely: (i) the unfairness

of TCP Congestion Avoidance, and (ii) the inability to identify the cause of packet loss. Various solutions suggested to overcome these issues are described. Finally, it is mentioned that though the proposed solutions address and resolve the problem at hand, they may not be applicable for all scenarios.

"I didn't think; I experimented." - Wilhelm Roentgen

3

Fairness Algorithms and Performance Implications

This chapter reports the simulation studies and test bed experiments conducted in connection with the first issue addressed by this thesis. The objective of the experiments is to show that although the fairness algorithms mentioned in Section 2.3.1 address and resolve the issues highlighted in Section 2.2.1, they tend to be harmful to connections traversing slow links and long thing networks.

Currently, there are a significant number of users connecting to the Internet through slow last-hop links, usually the 56Kbps modem links. And now, with the increase in the number of wireless devices, there is a growing number of users connecting to the Internet through Wireless WANs (W-WAN), often known as the Long Thin Networks (LTN).

In today's Internet, a 56Kbps modem link typically used by a laptop to access the Internet is considered as a slow link. One way propagation delay on slow links is around 50ms. The Long Thin Networks (LTNs) are called "Long" networks as they are characterized by large round trip times and are "Thin" since the networks usually have a low bandwidth. The common examples of LTNs are the Wireless WANs (W-WAN) like the GSM (Global System for Mobile communications), CDPD (Cellular Digital Packet Data) and Ricochet [4]. For a W-WAN like Ricochet the typical RTT value is around 500ms and the bandwidth is 24Kbps [4]. As can be seen, both LTNs and slow links are low speed links. However, unlike slow links, LTNs have relatively long and highly varying RTTs, which results in variable bandwidth delay products. Some LTNs like GSM offer a reliable wireless link by employing link level error recovery mechanisms. Therefore, increased link error rate would also result in increased RTT.

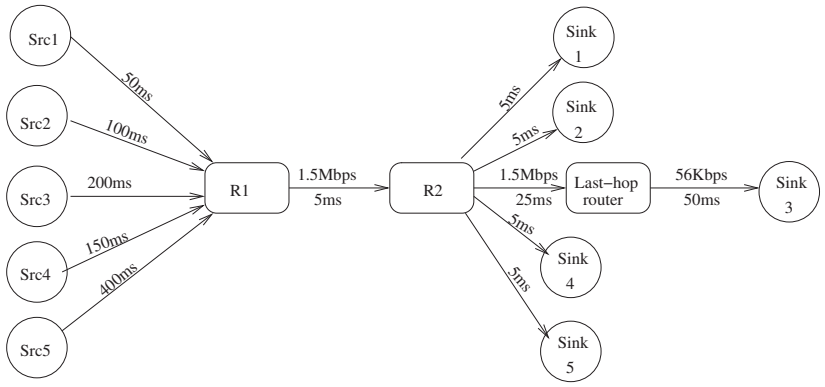
The slow links and LTNs are usually considered to be the last-hop links, connecting end user to the global Internet. That is, slow links or the LTNs are used as access links, with most part of the connection traversing through relatively high speed links. Hence the last-hop router to which the slow link or LTN is connected, may receive packets at a higher speed than it can forward on a low speed link. The last-hop router usually has a limited number of buffers configured for each outbound link. Sometimes even as low as a buffer size of 3 packets [3,4,19]. The last-hop router becomes a point of congestion and excess packets get dropped. This is an existing issue and many studies have been conducted [19–21] to see the implications of limited buffer size on the last-hop router.

As described above, it is not uncommon to have slow "access" links connecting to the Internet. We consider the following cases : (a) a slow link like a 56Kbps modem link used as an access link to connect to the global Internet and is a part of a long RTT connection; (b) a connection which has a long RTT because of the presence of a LTN as an access link in the path. In both the cases the TCP sender can only perceive a long RTT connection, but it is ignorant of the characteristics of the paths in the network. When the TCP sender is equipped with the policies proposed in [7,9,10] that are designed to improve fairness, it reacts by opening up the congestion window at a much higher rate. However, the connections described in the cases above only offer a long RTT but not high bandwidth. Hence, in these cases, the increased probing harms the connection. As mentioned earlier, the policies of [7, 9, 10] were designed so that long RTT connections get a fair share of the bottleneck bandwidth. However, these policies are not appropriate for slow links and LTNs, since they themselves are the bottlenecks in the network. In the following sections we evaluate the impact and identify ways to reduce the impact of the policies proposed in [7, 9, 10] on connections like the ones described above.

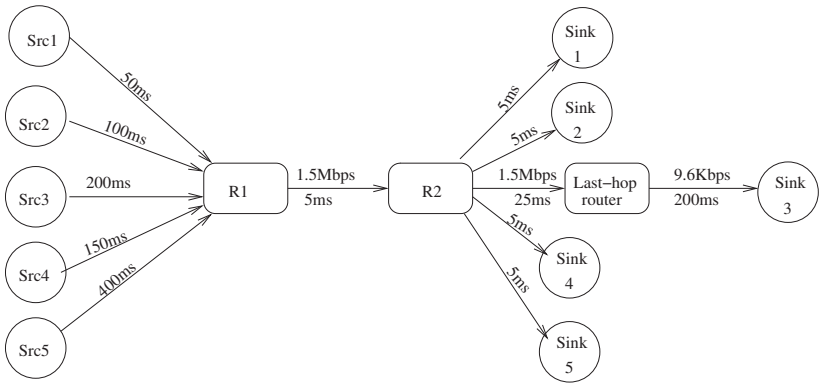
3.1 Simulation Setup

We use Network Simulator "ns" [22] to evaluate the impact of the proposed policies on slow links and LTNs. In this section, we describe the topologies used in our simulation study. The test configuration shown in Figure 3.1 was chosen as it has been used previously by the research community [7, 10, 23] in studies related to the

improvement of fairness. All the links unless specified have a bandwidth of 10Mbps. The path from source 3 to sink 3 represents a long RTT connection comprising a slow last-hop link of bandwidth 56Kbps and one way propagation delay of 50ms. Similar values were used in earlier studies to emulate a dial up access link [24]. The



(a) Test configuration 1 : Slow link



(b) Test configuration 2 : LTN

Figure 3.1: Simulation topology

bandwidth and propagation delay of the path connecting router R2 and the last-hop router are chosen to be 1.5Mbps and 25ms, respectively. The values were chosen in accordance with a study previously conducted on the last-hop router with limited buffer size [19]. We consider RTT of each connection to be twice the sum of all the

link delays along the path. The utilization of the link connecting router R2 and the last-hop router is kept close to 1. In the test configuration 2, the connection from source 3 to sink 3 represents a connection that traverses a long thin network with bandwidth 9.6Kbps and a constant one way propagation delay of 200ms. We call the connection that traverses slow link (resp. LTN) as the Slow link connection (resp. LTN connection).

3.2 Simulation Study

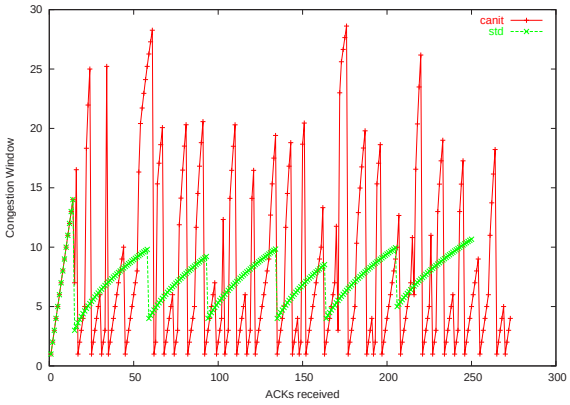
First, our interest is to see the possible impact caused by the CR, IBK and CANIT policies [7, 9, 10] on connections traversing slow links and LTN. Hence, we try to analyze the behaviour of those policies in the following subsection. A buffer size of 3 packets, on the last-hop router is a common configuration followed by Internet Service Providers (ISPs). We have chosen a buffer size of 3 packets in our studies since it is a common case [3, 4, 19]. The MTU (Maximum Transmission Unit) size is chosen as 296 bytes, in accordance with the recommendations in [3]. We conducted 20 independent simulation runs for each experiment and the duration of each simulation was 100s.

We evaluate the performance of the connections via two metrics, namely, goodput and packet losses incurred. These metrics enable us to identify the data that actually gets across to the receiver and the amount of data lost due to buffer overflow. In "subsection B", we study the impact of those policies by varying the buffer

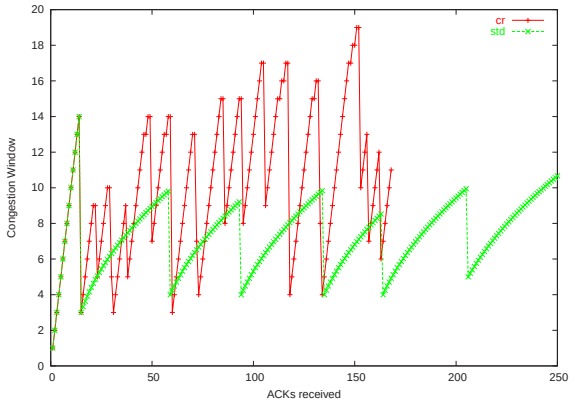
sizes from 3 to 40 packets and try to identify the ways to reduce the impact on the slow links and LTNs.

3.2.1 Behaviour of IBK, CR and CANIT Policies on Connections that Traverse Slow Links and Long Thin Networks

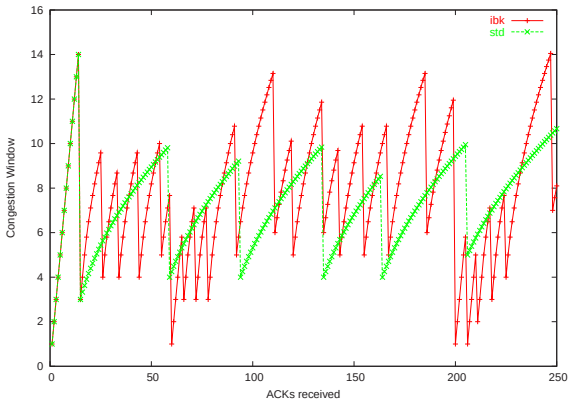
We are interested in seeing the amount of increase in congestion window, caused by each arriving ACK when different policies are employed.Figures 3.2 (a) (b) and (c) are plots of congestion window against the ACKs received, for slow link connection (test configuration 1) when CANIT, CR and IBK policies are used, respectively. IBK policy has to be selectively enabled on long RTT connections. Therefore, in all our experiments we enable the IBK policy only on the connections from source 3 to sink 3, source 4 to sink 4 and source 5 to sink 5. All the policies are compared with the behavior of the standard congestion avoidance algorithm. Figures 3.3 (a) and (b) are plots of congestion window against ACKs received, obtained by using test configuration 2. Fig 3.3 (a) shows the CR policy compared with the standard policy. Fig 3.3 (b) shows the variation in congestion window caused by the IBK policy compared with standard policy. During slow start all the schemes behave similarly, which is expected. However, during the congestion avoidance phase, the standard algorithm reacts to congestion by increasing its congestion window by *1/cwnd* on the receipt of every acknowledgement, while the other policies increase their congestion window at a much higher rate. In the congestion avoidance schemes that consider RTT as a parameter to increase the congestion window, there is a steep



(a) CANIT Vs Standard



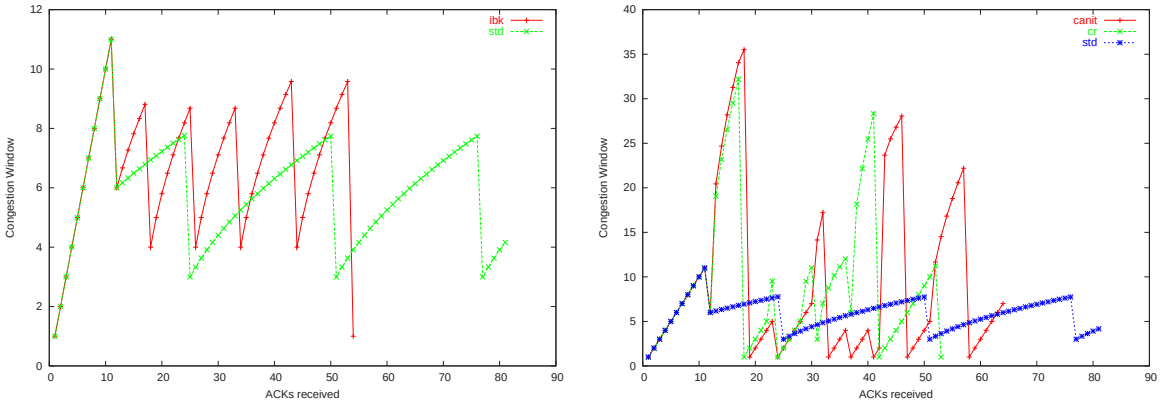
(b) Constant Rate Vs Standard



(c) IBK Vs Standard

Figure 3.2: Congestion window variation with arrival of ACKs for slow link connection

increase in the congestion window with the receipt of every acknowledgement. The reason for the steep increase is obvious : for instance, in CANIT , the congestion window is increased at the rate of $\frac{RTT}{NIT} \cdot \frac{1}{cwnd}$ on the receipt of every acknowledgement. We have considered the optimum value of NIT to be 30ms, as mentioned in [10]. The RTT of the connection could be as high as 500ms. Therefore, this algorithm would enable the TCP sender to inject many segments into the network on the receipt of every ACK. On connections with slow links or LTNs and limited last-hop router buffers, this could be very harmful. Figures 3.2 and 3.3 show a large number of retransmission timeouts, and congestion window being reset to 1. The effect is



(a) IBK Vs Standard (b) Constant Rate Vs Standard

Figure 3.3: Congestion window variation with arrival of ACKs for LTN connection

reflected in the number of losses and goodput of the connection as can be seen in Table 3.1 and Table 3.2.

In Constant Rate Policy, the congestion window is increased at the rate of $\frac{c \cdot RTT \cdot RTT}{cwnd}$ on the receipt of every acknowledgement. According to [7], values of 'c' less than 100 were best but smaller values of 'c' decreased the link utilization. Therefore, in our studies we consider the values of 'c' as 10. Higher values of c would only make the connection more aggressive [7]. The losses are relatively high and the goodput is lower than that achieved by the standard algorithm. The CR policy, even with a value of 'c' as small as 10, could prove to be very aggressive when the RTT is high, like in the cellular links. The impact is worse if a higher value of 'c' is chosen, since it would cause more packets to be injected into the network. In test configuration 2, we have assumed the LTN to have a typical value of 200ms for the propagation delay. However, in reality LTNs are known to have, long and varying RTTs, sometimes in the order of seconds. As can be seen in the above policies, higher values of RTT will only make the connection more aggressive, since more

packets will be injected into the network.

Policy	Goodput (Kbps)	Losses
STANDARD	16.54	8
IBK	12.68	45
CR	13.07	82
CANIT	1.86	304

Table 3.1: Performance comparison for test configuration 1

Policy	Goodput (Kbps)	Losses
STANDARD	7.62	10
IBK	6.90	21
CR	6.39	27
CANIT	2.77	62

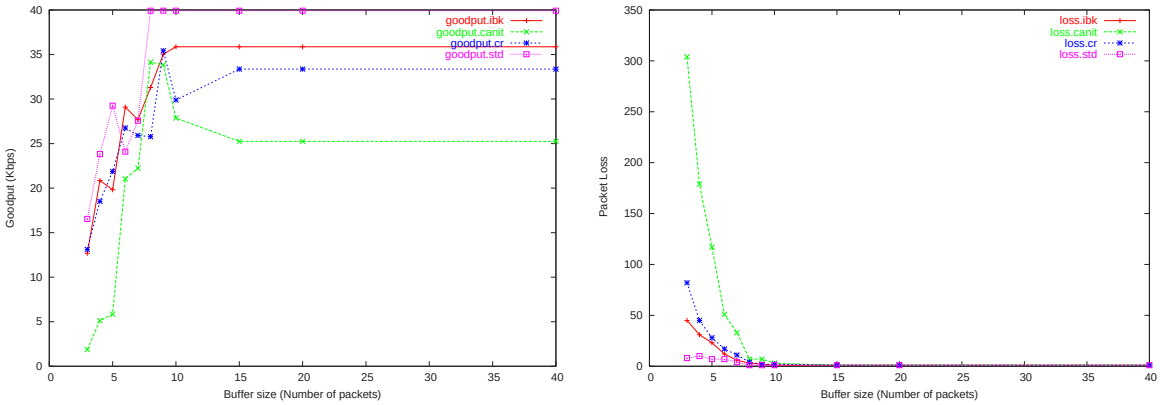
Table 3.2: Performance comparison for test configuration 2

It is seen that the losses incurred is relatively less in the case of "Increase-by-K" policy. According to the policy, congestion window is increased by a minimum of (K/cwnd, 1 segment) on the receipt of every acknowledgement. The limit ensures that the connections can only get as aggressive as slow start. As can be seen in the graph in Figures 3.2 (c) and 3.3 (a), the slope of IBK during the congestion avoidance phase resembles that of slow start.

3.2.2 Impact of Last-hop Router Buffer Size on Performance

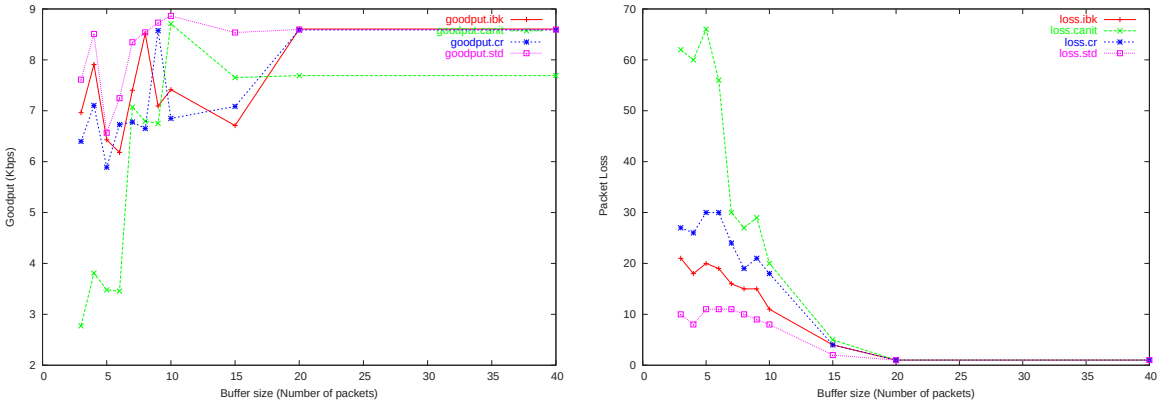
We have seen in the previous subsection that the RTT-aware fairness policies send out segments in bursts and buffer size of 3 packets for the last hop router is too small to absorb the bursts. In this subsection, we study the effect of having increased last-hop router buffer sizes. Figures 3.4 (a) and (b) are obtained using the test configuration 1, and depict the variation in goodput and losses respectively, on

varying the buffer size. Figures 3.5 (a) and (b) are obtained by using Test configuration 2 and depict the variation in goodput and losses incurred respectively, on varying the buffer size. Degradation in goodput and increased losses are seen,



(a) Goodput (b) Loss

Figure 3.4: Goodput and loss for slow link connection

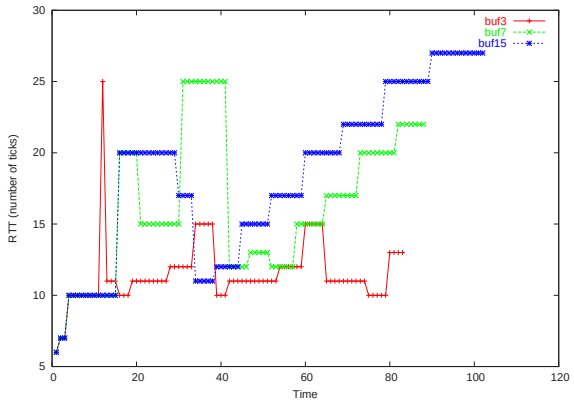


(a) Goodput (b) Loss

Figure 3.5: Goodput and loss for LTN connection

especially when the buffer sizes at the last-hop router are small. Large variations in goodput is seen, as the buffer sizes are increased. Sometimes, it is seen that the goodput drops as the buffer size is increased. This happens when losses occur at the end of a connection, causing a retransmission timeout. Similar observations were recorded in a study conducted previously [21]. The losses incurred due to the use

of the proposed policies are more than 2 to 3 times than that incurred by using the standard policy. As expected, the standard congestion avoidance policy performs much better with losses being very low and the goodput relatively high. With large buffer sizes, the traffic bursts can be absorbed and the use of the proposed policies do not have much impact. But increased buffer sizes introduces additional queuing delay and therefore increases RTT of the connection. The effect of increased buffer size on RTT for a LTN connection is shown in Figure 3.6. The RTT values for buffer sizes of 3, 7 and 15 are plotted. We see that the value of RTT is the least at a buffer size of 3.



(a)

Figure 3.6: RTT variation for different buffer size

3.2.3 Impact of Selectively Disabling the Policies on Performance

In the test configuration 1, the bottleneck link for the connection between source 3 and sink 3 is 56Kbps, whereas for all the other connections the bottleneck link is 1.5Mbps. The proposed policies [7,9,10] are designed to ensure a fair allocation of the

shared link (in this case, 1.5Mbps) to all the competing connections. However, it can be seen that irrespective of the policies used to improve fairness, the connection from source 3 to sink 3 can only have a maximum throughput of 56Kbps. It is the same case for the connection with LTN, which can only a have a maximum throughput of 9.6Kbps. Hence it is not meaningful in trying to get a fair allocation of the shared link for connections with slow links and LTNs. The policies are not appropriate to such connections and need not be applied. We have seen in earlier subsections, that such policies which are intended to provide a fair share will only end up harming the connections. Therefore, in the next experiments we selectively disable the policies for connections traversing slow links or LTNs. In other words, the standard congestion avoidance policy is used on connections traversing slow links or LTNs, while all the other connections use the alternate policies, namely IBK, CR or CANIT. In Tables 3.3 and 3.4, column 3 indicates the goodputs for the Slow Link/LTN connection when the corresponding policy is disabled for the Slow link / LTN connection. (i.e, when the the standard policy is used on the slow link/LTN connection while all other connections are enabled with CR, IBK or CANIT policies). As mentioned earlier, IBK policy is selectively enabled for long RTT connections . We use a limited buffer size of 3 packets at the last-hop router. We notice that the standard policy gives the best performance for slow link and LTN connections. However, as we have seen in Table 3.1, with standard policy being enabled on all the connections, the throughput of the slow link connection is 16.54Kbps which is greater than the values in column 3 of Table 3.3, when competing connections use aggressive congestion avoidance schemes. The same applies to the goodput of the LTN connection. The goodput of

the LTN connection when all the competing connections use the standard algorithm is 7.62 Kbps as in Table 3.2, which is greater than the values in column 3 of Table 3.4. This is expected since the competing connections are enabled with alternate policies and can open up their windows at a much higher pace compared to the connections using standard congestion avoidance policy.

	Goodput (Kbps)	
Policy	Enabled on all connections	Disabled only for slow link
IBK	12.68	15.26
CR	13.07	13.98
CANIT	1.86	13.94

Table 3.3: Performance of slow link connection when policies are selectively disabled

	Goodput (Kbps)	
Policy	Enabled on all connections	Disabled only for slow link
IBK	6.90	7.44
CR	6.39	7.58
CANIT	2.77	7.08

Table 3.4: Performance of LTN connection when policies are selectively disabled

3.2.4 Impact of Advertising a Limited Receive Window on Performance

The recommendations in [3] suggest that the hosts which are directly connected to low speed links could advertise a limited receive window in order to reduce or eliminate the losses at the last-hop router. In the next set of experiments, instead of selectively turning off the policies for the slow link and LTN connections, we allow the congestion window to increase according to IBK, CR or CANIT policies but restrict the amount of data injected into the network by advertising a smaller receive window in accordance with the recommendations in [3]. The advertised

receive window is chosen to have a value slightly more than the pipe capacity of the low speed link inclusive of the last-hop router buffer size. In a similar study conducted on slow wireless links [20], it was found that a limited advertised receive window of 2KB was optimum. Hence, we have chosen the value of 2KB in our studies. As we can see in Table 3.5 and Table 3.6, there is a marked decrease in the number of losses incurred. And for the connection using CANIT, the goodput is increased since the number of losses incurred is vastly decreased. However, there is a marginal decrease in the goodput in other cases, in spite of decreased losses. A smaller advertised receive window restricts the amount of data injected into the network by the TCP sender. Therefore, the number of packet losses incurred due to buffer overflows at the last-hop router is reduced. However, imposing a limit on amount of data transmitted, also limits the sender from transmitting new segments during the fast recovery phase [21]. Besides, the competing connections are enabled with aggressive policies. These are the reasons for the decrease in goodput.

	Goodput (Kbps)		(Losses)	
Policy	No Limit	Limited Rwnd	No Limit	Limited Rwnd
IBK	12.68	11.78	45	21
CR	13.00	12.92	82	16
CANIT	1.86	11.72	304	55

Table 3.5: Performance of slow link connection with a limited receive window

	Goodput (Kbps)		(Losses)	
Policy	No Limit	Limited Rwnd	No Limit	Limited Rwnd
IBK	6.90	6.71	21	17
CR	6.39	6.89	27	14
CANIT	2.77	5.65	62	24

Table 3.6: Performance of LTN connection with a limited receive window

3.3 Test-bed Experiments

We conducted experiments in a controlled network environment in order to study the behavior and impact of fairness policies on the performance of slow links and LTN. The objective of the test-bed experiments was to examine the impact of fairness algorithms in a controlled network environment with real TCP/IP stack implementation and live GSM network. The test bed was setup to conduct experiments in two specific scenarios - (i) a 56kbps modem link with parameters typical to dial-up connections, and (ii) a client connecting to the testbed through SINGTEL GSM network at 9.6kbps.

3.3.1 Test Configuration

A controlled network environment was set up to carry out the tests as shown in Figure 3.7. All the sources and routers were running FreeBSD 4.5. Bandwidth and delay was controlled using dummynet [25]. The bandwidth was limited to 1.5Mbps at the Bandwidth Controller, to emulate the bottleneck bandwidth. Delay Box 1 was used to introduce a delay of 50ms for src1, 30ms for src2 and 50ms for src3. Delay Box 2 set a delay of 50ms for sink1, 30ms for sink2 and 50ms for the last-hop router. Therefore the RTT for the connection from src1 to sink1 was 200ms and for the connection from src2 to sink2, the RTT was 120ms. The last-hop router was setup as a PPP server, and sink3 was setup as a PPP client. To conduct tests on slow links, the PPP connection was setup with a bandwidth of 56Kbps, and 75ms delay. For LTN test, the GSM connection had a bandwidth of 9.6 Kbps, and

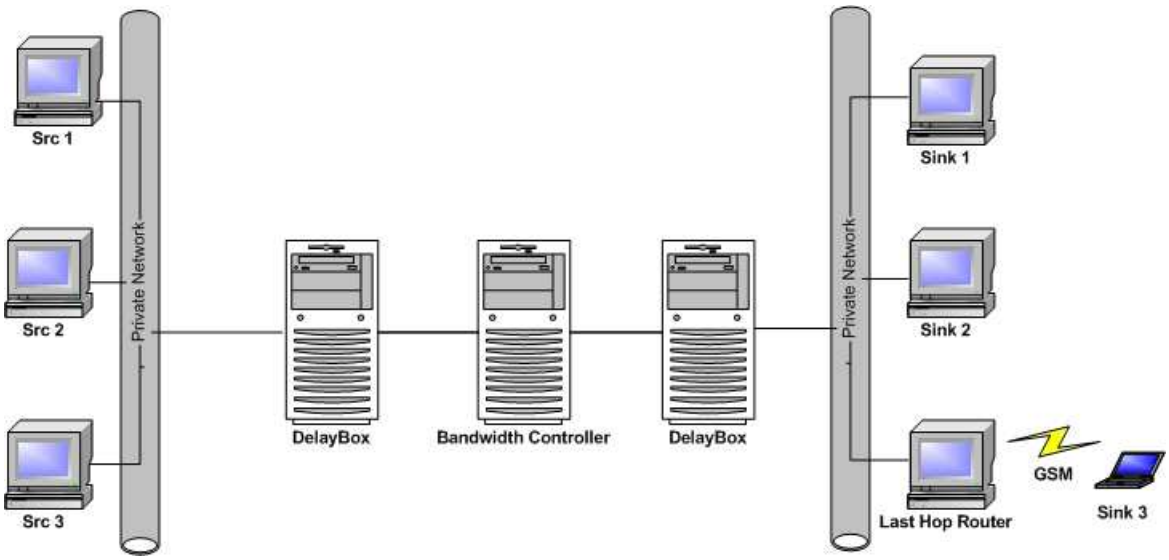


Figure 3.7: Test configuration

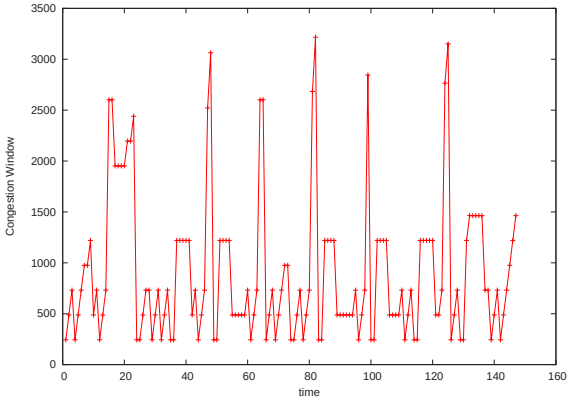
approximately 400ms delay. A Nokia D211 multimode radio [26] card was used as the GSM modem. The RTT for the connection from src3 to sink3 using slow link was 350ms, and the RTT for connection when using the LTN was around 900ms. It was ensured that the utilization of the bottleneck bandwidth was close to 1. The link under study, slow link or LTN as the case may be, is from src3 to sink3.

Figure 3.8: Nokia D211 PCMCIA multimode radio card

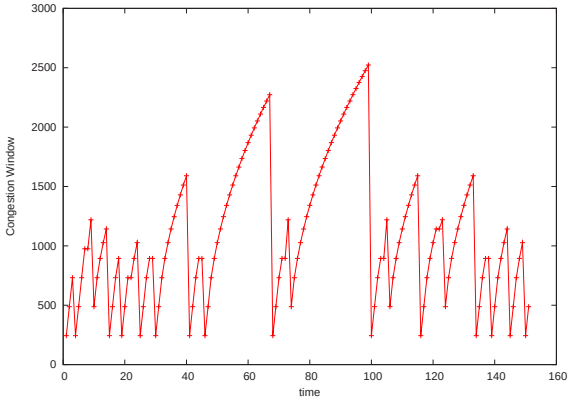
3.3.2 Impact of IBK, CANIT and CR Policies on Slow Link and LTN Connections

Each of these policies attempt to ensure fairness by increasing cwnd at a faster rate for connections with long RTTs. The effect of the standard congestion avoidance

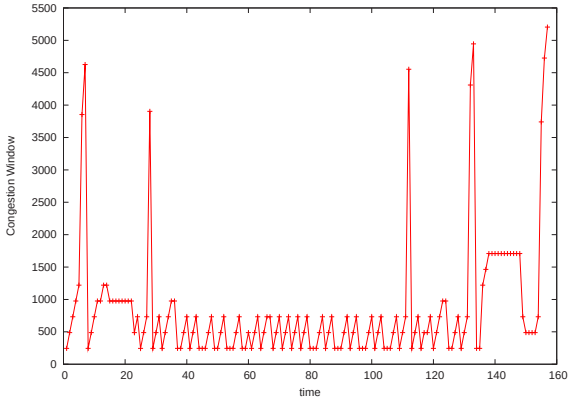
policy, CANIT, IBK and CR on cwnd of slow link connections is shown in Figures 3.9 (a) (b) and (c) respectively. As can be seen, the cwnd for CANIT and CR policies exhibit very steep increases. This is because both CANIT and CR, are very sensitive to the RTT of the connection. The long RTT of the slow link causes both policies to increase cwnd at a much faster rate than the standard and IBK policies do. IBK increases the cwnd in a very similar fashion to the standard algorithm, but at twice the rate. Table 3.7 shows the goodput and error rate of the slow link and the LTN



(a) CANIT



(b) IBK

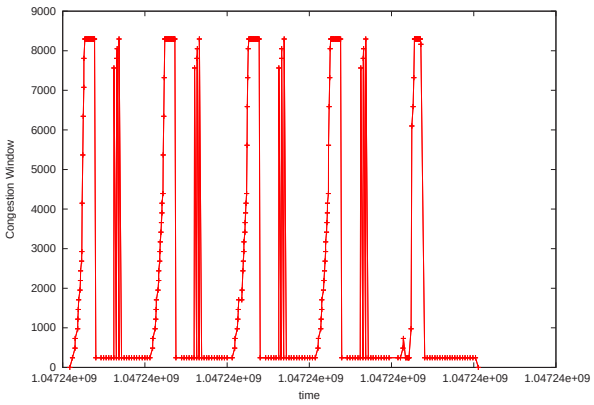


(c) CR

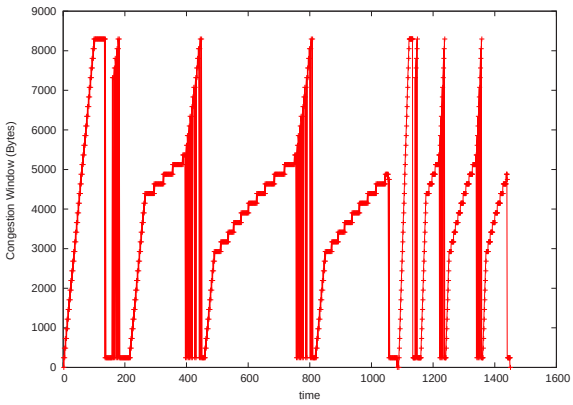
Figure 3.9: Variation of congestion window for slow link connection

connection, when the various policies are implemented on all connections sharing

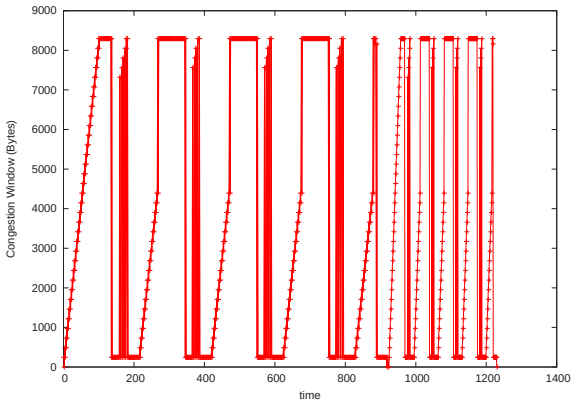
the bottleneck link. The goodput for the connection when using the standard policy is only 1 KBps, even though the bandwidth is 7 KBps (56Kbps), the reason being the other TCP flows that were competing with this connection for bandwidth and buffer space at intermediate gateways.



(a) CANIT



(b) IBK



(c) CR

Figure 3.10: Variation of congestion window for LTN connection

Due to the limited buffer space, packets are dropped at the last hop-router. This causes TCP to go into slow start frequently, as can be verified by cwnd diagrams in Figure 3.9. Policies that rely on RTT as a measure of bandwidth of the connection,

Policy	Slow Link		LTN	
	Goodput (Kbps)	Loss	Goodput (Kbps)	Loss
Standard	8.08	6.2	5.64	3.8
IBK	7.32	11.5	5.25	6.4
CANIT	6.92	13.2	2.65	19.5
CR	4.98	58.6	2.56	18.3

Table 3.7: Performance of slow link and LTN for various congestion avoidance policies

like CANIT and CR, increase cwnd proportionaly to RTT, causing more buffer overflows at the last-hop router than the standard and IBK policies.

3.3.3 Impact of Last Hop Router Buffer Size on Performance

Most of the retransmission timeouts occur due to buffer overflows at the last hop router. By intuition, an increase in the last hop router buffer size should lead to better throughput. This can be verified in the results obtained in Table 3.8 and 3.9. However the increase in buffer size increases the queuing delay, which leads

Policy	Goodput (Kbps)		
	Buffer-3	Buffer-10	Buffer-20
Standard	8.08	31.35	37.85
IBK	7.32	30.65	36.56
CANIT	6.92	31.57	37.12
CR	4.98	9.2	27.12

Table 3.8: Effect of buffer size on goodput of the slow link connection

to a longer RTT. For a buffer size of 3 at the last hop router, the avg. RTT was 659.7ms; for a buffer of 10 packets the avg. RTT was 686.2ms; and for buffer of

Policy	Goodput (Kbps)		
	Buffer-3	Buffer-10	Buffer-20
Standard	5.64	5.91	6.04
IBK	5.25	5.79	5.86
CANIT	2.65	2.85	3.44
CR	2.56	2.68	3.42

Table 3.9: Effect of buffer size on goodput of the LTN connection

20 packets the avg. RTT was 855.7ms.This longer queuing delay could adversely effects user interactive applications like telnet or ssh.

3.3.4 Impact of Receiver's Advertised Window on Performance

For the slow link, the throughput is much lower than the capacity of the link because of losses at the last-hop router, causing retransmission timeouts. Therefore a lower receive window leads to lesser packets being dropped at the last hop router, thus increasing the goodput. This can be verified from Table 3.10.

Policy	Goodput (Kbps)		Loss	
	Rwnd(1 KB)	Rwnd(8 KB)	Rwnd(1 KB)	Rwnd(8 KB)
Standard	10.05	8.08	1.6	6.2
IBK	10.8	11.5	0.7	11.5
CANIT	10.86	6.92	0.7	13.2
CR	10.01	4.98	1.7	58.6

Table 3.10: Effect of receiver window on slow link connection

Policy	Goodput (Kbps)		Loss	
	Rwnd(1 KB)	Rwnd(8 KB)	Rwnd(1 KB)	Rwnd(8 KB)
Standard	4.41	5.64	0.2	3.6
IBK	4.46	5.25	0.0	6.4
CANIT	4.41	2.65	0.0	19.5
CR	4.42	2.56	0.0	18.3

Table 3.11: Effect of receiver window on LTN connection

For the LTN however, the capacity of the link is only 1.2KBps (9.6 Kbps). The utilization of the link is quite high. The standard and IBK policies are quite conservative, and a low receive window restricts them from sending enough data to fill the pipe, causing a lower throughput. It does, however, manage to improve the throughput for CANIT and CR policies, which are relatively aggressive.

3.3.5 Impact of Selectively Disabling Fairness Policies on Slow Links and Long Thin Networks

The fairness policies were disabled on the slow link or LTN connection, but were enabled on the competing connections. As can be seen from Table 3.12 and 3.13, the performance of slow link connection and LTN connection is improved when the fairness policies are disabled, even when competing connections are using aggressive fairness policies.

	Goodput (Kbps)	
Policy	Enabled on all connections	Disabled only for slow link
IBK	7.34	7.96
CANIT	6.92	7.96
CR	4.98	8.48

Table 3.12: Performance of slow link connection when policies are selectively disabled

	Goodput (Kbps)	
Policy	Enabled on all connections	Disabled only for slow link
IBK	5.25	5.72
CANIT	2.65	5.76
CR	2.56	5.11

Table 3.13: Performance of LTN link connection when policies are selectively disabled

3.4 Recommendations

Our studies were focused to show that the algorithms to improve fairness have adverse effects on connections traversing either slow links or LTNs. We have argued that it is not appropriate to apply the fairness algorithms to connections that traverse slow links or LTNs. We have also shown that, increased buffer sizes in the last-hop router absorb the bursts but only at the cost of increased delay. Advertising a limited receive window on the LTN/slow link connection reduces the number of packet losses but harms the fast recovery mechanism. Besides, the competing connections enabled with the fairness policies are more aggressive as they can inject more data into the network compared to the slow link/LTN connection with limited advertised receive window. Hence, advertising a limited receive window cannot mitigate the impact. It is seen that the impact could be reduced by selectively turning off the policies for slow link/LTN connections.

3.5 Summary

This chapter delineates the first issue addressed by the thesis viz., Effect of algorithms that improve fairness of TCP congestion avoidance on the performance of slow links and long thin networks. This chapter started by describing the impact of fairness algorithms on connections traversing slow links and long thin networks. Simulation experiments are conducted to show that the fairness algorithms have adverse effects on slow link or long thin networks. Test bed experiments are conducted

and the results show behavior similar to the one identified in simulations. Finally, the chapter is concluded with recommendations for performance improvement.

"The best way to escape from a problem is to solve it."

- Alan Saporta

4

SPEP: Secure Performance Enhancing Proxy

Several studies have been conducted and various schemes have been proposed as discussed in Section 2.3.2 to address the suboptimal performance of TCP in a hybrid wired-wireless network. However, the proposed solutions were designed oblivious of security considerations and therefore fail to work in secured environment. In this chapter, we discuss the limitations of the existing schemes to function in a secured environment and propose an innovative mechanism, which we call Secure Performance Enhancing Proxy (SPEP), to address the problem of enhancing TCP performance over wireless networks while preserving the end-to-end semantics of TCP as well as ensuring end-to-end security.

With the proliferation of wireless hot spots, cellular networks and the antici-

pated merger of Wireless Wide Area Networks (WWAN) and Wireless Local Area Networks (WLAN), wireless networking is becoming a new way of gaining access to the Internet. The wireless link is known to be unreliable and improving the performance of applications over such links continues to be a concern. Most of the solutions previously proposed to address this problem, e.g., [13, 14, 18, 27–29] are designed oblivious of security considerations and violate the end-to-end semantics of TCP. Meanwhile, Internet applications such as e-commerce, business-to-business communication, banking etc. demand a secured environment for communication and conducting transactions. Achieving improved performance together with ensuring end-to-end security necessitates the co-existence of security mechanisms such as IPSEC and performance enhancing solutions. RFC 3135 [3] is a survey of existing performance enhancing proxies that describes the various approaches for improving TCP performance and their implications. The draft mentions that its authors are unaware of the existence of any PEP which provides support for IP Security. This RFC also mentions the research underway to change the IPSEC implementation to accommodate PEP. However, changing the IPSEC implementation to support PEP requires wide scale changes in the network besides introducing more complexity to the security architecture. The real challenge lies in designing a scheme which provides both end-to-end security and improved TCP performance without introducing complexity in the network.

4.1 Related Work and Issues

Several approaches have been proposed to enhance the performance of TCP over wireless links. In this section, we categorize some of these approaches based on their conformance to the principles of end-to-end security and discuss the limitations of various schemes. We also look at some attempts in the area of IP security which are designed to accommodate the existing performance enhancement solutions.

A. Performance enhancement schemes that cannot co-exist with end-to-end IP Security

Split connection approaches such as I-TCP [13], MTCP [14] propose to split the connection between the Fixed Host (FH) and Mobile Host (MH) at the intermediate base station (BS). This would necessitate the BS to maintain TCP state information of both the connections. The split connection approaches propose to have either TCP or a protocol optimized for wireless links for the connection between BS and MH. Though the schemes achieve performance improvement, they violate end-to-end TCP semantics and prevent the use of end-to-end IPSEC.

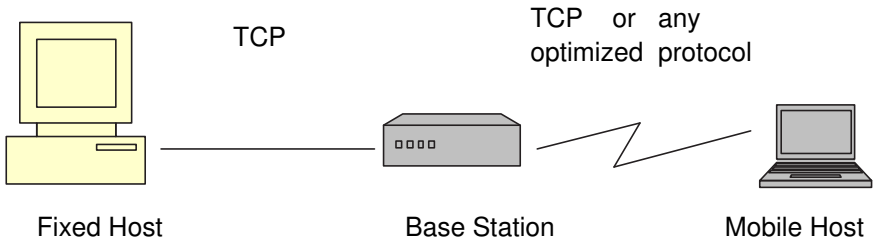


Figure 4.1: Split-Connection approach

Other approaches to improve the performance of TCP e.g., SNOOP [18], M-TCP [27], WTCP [28], ELN [29] preserve the end-to-end semantics of TCP. However, all the above mentioned approaches require base station mediation and are based on the assumption that TCP headers are readable by the base station. Therefore the schemes can function only in the absence of end-to-end IPSEC.

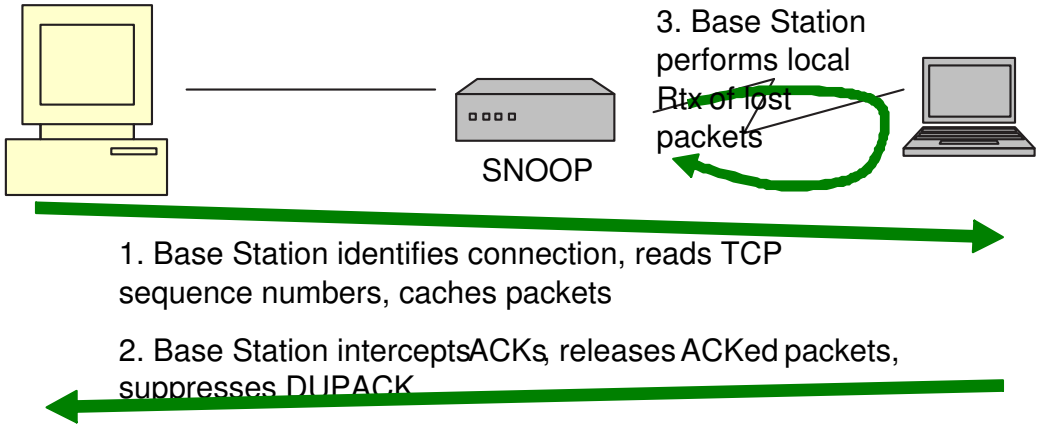


Figure 4.2: Snoop approach

TCP HACK [16] is another approach suggested to distinguish between corruption and congestion losses that preserves true end-to-end semantics. However, the inherent nature of TCP HACK renders it ineffective in an end-to-end IPSEC environment. A TCP HACK-enabled mobile node checks the received packet to determine if the corrupted portion is the TCP payload or the TCP header. If the corrupted portion is the TCP payload and the TCP header is intact, the packet is considered lost due to corruption and action is taken accordingly. However, when encrypted packets get corrupted, the decryption of corrupted packets results in the whole packet being garbled. This is not the case if encryption is carried out using stream ciphers in the Output Feedback mode or the Cipher Block Chain mode.

However, such modes of stream ciphers are vulnerable and necessitate the use of authentication for integrity check [30]. When integrity check is enabled, the corrupted packets are dropped by the IPSec protocol. Hence, TCP HACK cannot work with IPSEC.

B. Performance enhancement schemes that can co-exist with End-to-End IP Security

Some schemes [15, 31, 32] have been proposed that comply with the end-to-end principles. Freeze-TCP [15] is one such solution that can co-exist with IPSEC. However, Freeze-TCP provides a solution only for TCP performance degradation caused by frequent and/or long disconnections. Freeze-TCP does not address the problem of distinguishing congestion losses from corruption losses. The proposed SPEP approach complements the Freeze-TCP solution and Freeze-TCP can easily be accommodated with SPEP.

Delayed Duplicate-Acknowledgement [31] is a proposal to improve the performance of TCP over wireless links by delaying the third and subsequent duplicate acknowledgements if the packet loss is determined to be due to corruption. The scheme maintains end-to-end semantics and can function in a secured environment. However, the scheme has some open issues regarding the method to differentiate the nature of packet loss.

Another approach [32] explores the possible use of congestion avoidance techniques (CAT) as loss predictors to distinguish between congestion and corruption loss. The congestion avoidance techniques discussed in [32] are delay based. The

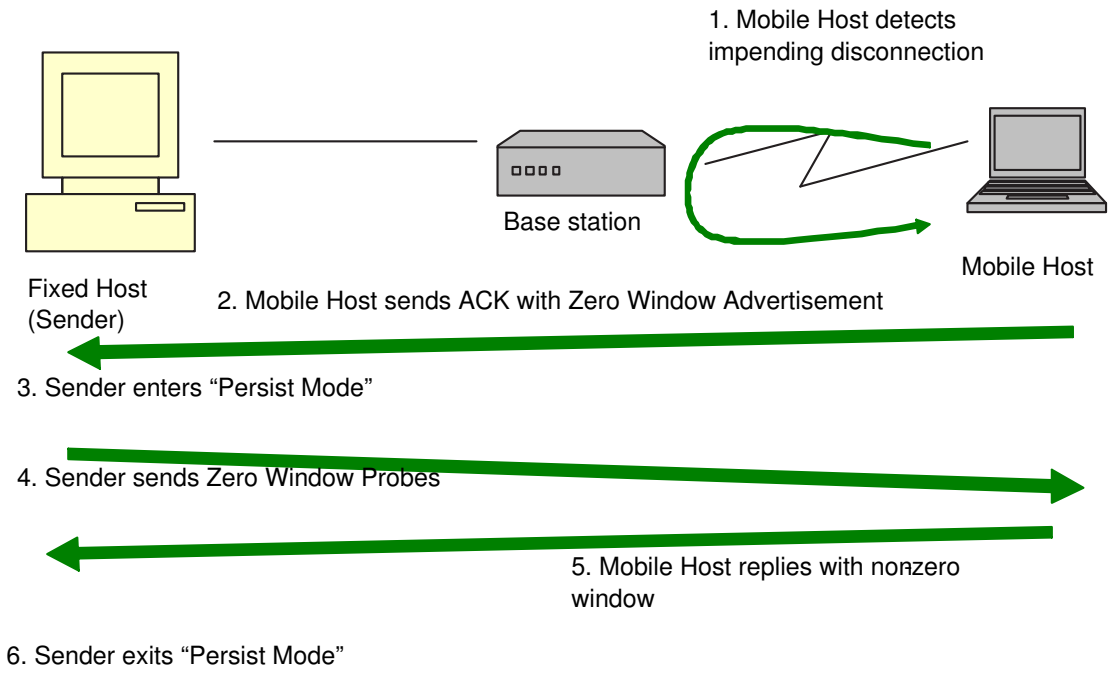


Figure 4.3: Freeze-TCP approach

decision to increase/decrease the congestion window is based on the size of current congestion window and RTT. The approach of using CAT as loss predictors can co-exist with IPSEC. However, the study [32] shows that the CAT based loss predictors can correctly identify congestion losses only if congestion losses are preceded by a "long" queue build-up, otherwise its futile to use CAT as a loss differentiation mechanism.

C. Novel Security schemes that can accommodate existing performance enhancement schemes

This subsection discusses some solutions that address the issue of inter working and coexistence of IPSEC and Performance Enhancing Proxies (PEP). In an attempt to provide an architecture for inter-working of PEP and IPSEC, Assaf et

al [33] propose the use of an intelligent switch at the PEP (intermediate nodes). The approach is straight-forward, the switch identifies if the packets are encrypted, which can be done easily by reading the unencrypted IP header information. The PEP normally handles the unencrypted packets while the encrypted packets bypass it. Encrypted packets are handled by PEP if the PEP happens to be a part of the security association. The above proposal only provides an architecture for the inter-working of IPSEC and PEP. With this approach, the applications can choose between security and performance, but both are not obtainable together.

SSL, as proposed by Netscape and later standardized by IETF as TLS [34], is a transport layer mechanism that provides data security. The suggested mechanism encrypts data, but not transport layer headers, such as those for TCP. Since the transport layer headers are now in plaintext, the intermediate nodes can access or modify the transport layer headers; thereby the performance related issues can be resolved. However, it is unacceptable to have TCP headers in plaintext due to security concerns as described later in this section. Suggestions were also made to use SSL/TLS with IPSEC in order to protect the header information. The use of SSL/TLS with IPSEC is not a good solution because (a) a lot of overhead is introduced because the packets have to undergo encryption/decryption process twice once at the SSL layer and then at the IPSEC layer, and (b) PEP cannot function as IPSEC encrypts the TCP headers.

Transport Friendly ESP (TF-ESP) or Modified ESP (M-ESP) [35] proposed a modification to the ESP header to accommodate the necessary TCP header infor-

mation in the ESP header outside the scope of encryption. The mechanism proposed that the unencrypted TCP header information in ESP should be authenticated for integrity. Although this method addresses the performance issues, it exposes enough information to make the connection vulnerable to security threats.

In the previous two approaches, exposing the transport layer header information was considered to be hazardous since it would facilitate the chances of a probable plaintext attack. As can be seen in [36], knowledge of full blocks of plaintext is not needed for cryptanalysis. The IP and the transport layer headers provide ample probable plaintext information that can be used to drive a cracking engine. Most fields of the TCP header are predictable. The only random parts and hard to predict fields of the TCP header are the sequence number field, the client's port number field and the window advertisement field. The previous two approaches make it easier for the attacker by exposing this vital information also.

The Multilayer IPSEC Protocol (ML-IPSEC) [35] proposed a multi-layer encryption scheme. The IP datagram payload is divided into zones; each zone has its own security association and protection mechanisms. For instance, the TCP data could be in one zone, using end-to-end encryption with the keys only shared between end-hosts. The TCP header could be in another zone with security associations between the source, destination and a few trusted nodes. The trusted nodes can decrypt the transport layer headers to provide the performance enhancements. This mechanism ensures security and can accommodate existing performance solutions. Though the requirements are satisfied, as can be seen, there is quite a lot of

complexity involved in key exchange and management. Also, the assumption that intermediate nodes are 'trustworthy' may not be acceptable.

4.2 The SPEP Approach

The SPEP approach addresses the problem in a different way. As discussed in Section 4.1, the previous approaches either totally ignore security considerations or make modifications in the security architecture to run TCP performance improvement solutions. Modifications in the security architecture are done with the assumption that the TCP header information is vital for the proper functioning of PEP. In the TF-ESP approach the TCP header information is included in the ESP header in plaintext but authenticated, and in ML-IPSEC the 'trusted' intermediate nodes capable of decrypting TCP headers are included in the security association.

The SPEP approach is a paradigm shift and attempts to provide security and performance with a minimal increase in complexity. We realize that the actual cause for the problem is the fact that performance enhancement proxies are implemented at the transport layer and the security is implemented at the network layer. SPEP is a secure performance enhancing proxy implemented at the network layer; it proposes a novel way to implement PEP rather than change the IPSEC implementation to support PEP.

4.2.1 SPEP Overview

The SPEP approach decouples the loss detection and loss distinction mechanism from the loss recovery mechanism. The mechanism for loss detection and distinction is implemented at the network layer. The loss recovery is done at the transport layer by using the information about the nature of loss obtained from loss detection and distinction mechanism at the network layer.

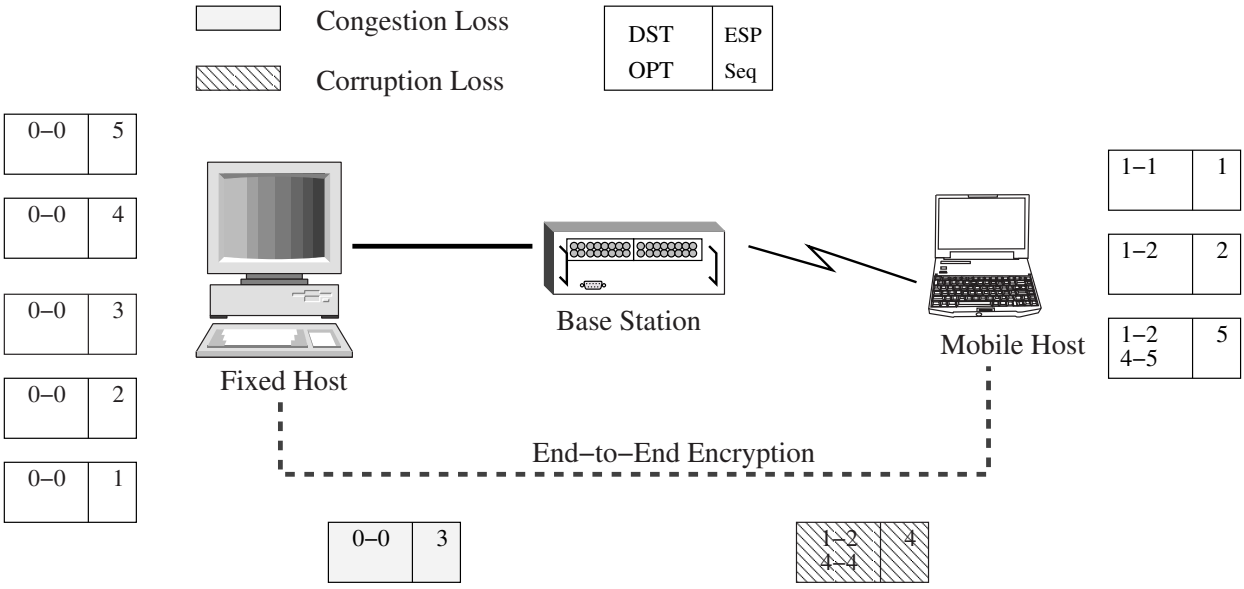


Figure 4.4: The SPEP approach

A. Loss Detection and Loss Distinction

The SPEP approach comprises two components one located at the base station and the other at the mobile node. The SPEP base station component facilitates in distinguishing between congestion and corruption losses based on the fact that losses that occur in the path between the fixed host and the base station are caused by

network congestion and the losses that occur in the path between the base station and the wireless node are caused by corruption in the wireless link. The SPEP scheme requires an additional option field in the IPv6 header for carrying the blocks of sequence numbers received in-sequence at the base station. The option field is initialized to 0 at the fixed host. The base station keeps track of the blocks of sequence numbers received in-sequence. The base station then populates the option field in each packet to reflect the blocks of in-sequence packets received. Figure 4.4 shows packets 1-5 transmitted from the fixed host. The portion of the packet initialized as 0-0 is the option field and the other portion contains the sequence numbers. Packets 1, 2 are received in sequence at the base station and the option field of packet 2 is populated as 1-2, indicating that packets 1, 2 were received in sequence. Packet 3 is lost, the next packet received by the base station is 4 and option field of packet 4 is populated appropriately with 1-2, 4-4 indicating that it received a block of packets from 1-2 in sequence followed by a block of packets 4-4 in sequence. Following the same logic, when packet 5 arrives, the option is populated as 1-2, 4-5. It may be inferred that the packets that do not feature in the blocks of in-sequence numbers that are received at the base station are lost due to congestion.

The base station forwards the packets 1,2,4,5 to the mobile node. Suppose packet 4 is lost due to corruption over the wireless link, the mobile node receives packets 1, 2 followed by packet 5. On receipt of packet 5, the SPEP mobile node component realizes that packets 3 and 4 are lost. The packet 5 contains the option field with values 1-2, 4-5. The SPEP mobile node component searches for sequence numbers of the lost packets 3, 4 in the option field of the received packet 5. Packet

3 is not part of the blocks of in-sequence numbers received at the base station. This implies that packet 3 was not received even by the base station. Hence it is concluded that it was lost due to network congestion. Packet 4 features in the block of in-sequence numbers received at the base station. This implies that packet 4 was received by the base station. However, packet 4 was not received by the mobile node. Hence packet 4 must have been lost due to corruption in the wireless link.

The SPEP approach explained above is capable of distinguishing the losses by tracking only the data packets and does not require the acknowledgements (ACK) to be tracked by base station. Hence the approach can co-exist with the end-to-end security principles. The SPEP approach is not affected by packet-reordering as explained in Section 4.3.1.

B. Loss Recovery

The SPEP mobile node component detects the nature of loss and stores the information for each lost packet in the TCP reassembly queue. TCP invokes the loss recovery mechanism by sending duplicate acknowledgements for the next expected packet. The vital information regarding the nature of loss determined by the SPEP component is conveyed to the sender by setting a bit in the TCP header of the ACK packet. A mechanism like Explicit Loss Notification (ELN) implemented using the reserved bit in TCP header conveys to the sender about the corruption loss. The sender then retransmits the packet lost due to corruption without invoking congestion control.

4.2.2 SPEP Design Considerations

The implementation of the SPEP approach discussed in the previous section necessitates the following requirements:

- a means for the SPEP base station component to uniquely identify the flow;
- a monotonically increasing sequence number to uniquely identify each packet in the flow;
- an additional option field in each packet to carry the blocks of in-sequence numbers.

As mentioned earlier, the SPEP architecture is based on IPv6 and leverages on the features offered by it. The IPv6 header as shown in Figure 4.5 consists of an encrypted portion which includes the TCP header and data, and an unencrypted portion which includes the Encapsulated Security Payload (ESP) header, Destination Option (DSTOPT) header and the IPv6 header [37]. The IPv6 header has a 24-bit flow label field meant to be used by a source to uniquely label packets belonging to a particular flow which requires special handling at the intermediate routers. The flow label along with the source address can be used to uniquely identify a flow at an intermediate router, thereby satisfying the first requirement by SPEP.

IPv6 mandates the use of IPSEC for security and supports extension headers for purposes such as encryption and authentication of the IPv6 payload. The end

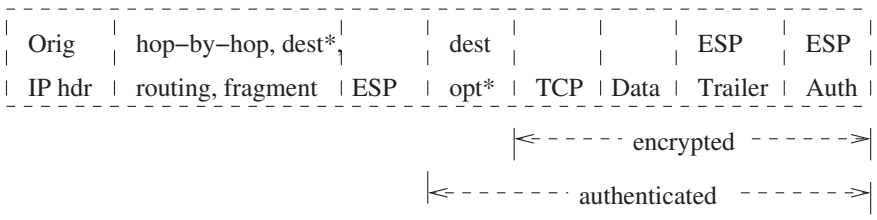


Figure 4.5: IPv6 header

hosts opting for end-to-end security should first establish a security association. A Security Association (SA) is a simplex connection that provides security to the data carried by it. The end hosts should contain mode, protocol, algorithms and keys required by the SA. In addition each end host should have a Security Policy Database (SPD). The SPD determines the processing required for each packet being sent out against discarding, applying or bypassing IPSEC. Packets can be classified based on a variety of selectors. The SPD can give a very fine-grained control so as to use a separate SA for every TCP connection [38]. Finally, the Authentication Header (AH) [39] or ESP [40] protocol can be used in the SA to provide confidentiality and/or authentication, data integrity and anti-replay services. The ESP Header has a 32-bit field which contains a counter value that increases by one for each packet sent out in a particular SA. Thus, the second requirement of SPEP as identified in the beginning of the section is satisfied.

The IPv6 header provides extension headers to carry optional Internet layer information. The DSTOPT header is an extension header of IPv6 used to carry optional information that needs to be examined by the packet's destination node(s) [37]. The DSTOPT header contains a variable length options field to carry Tag-

Length-Value (TLV) encoded options. Hence, the third requirement of SPEP can be satisfied by using the options field of DSTOPT header.

4.2.3 SPEP Implementation Description

The SPEP implementation was done in FreeBSD 4.5. The availability of a stable stack for IPv6 and IPSEC was the primary reason for choosing FreeBSD. The SPEP component at the base station performs the following three functions :

- uniquely identifies a flow using the <source address, flow label, destination address> triplet;
- keeps track of the ESP sequence number to check if the received packet is the next expected packet;
- updates the option field in the DSTOPT header as explained in the Section 4.1.

The SPEP mobile node component comprises loss detection and distinction mechanism and a loss recovery mechanism. The loss detection and distinction mechanism uses the algorithm described below.

The algorithm keeps track of the next expected ESP sequence number. If the received ESP sequence number is not equal to the next expected sequence number then it is inferred that some packets are lost. The difference between received and expected ESP sequence numbers gives the number of lost packets. The lost sequence

```
If ( recvd_esp_seq != expected_esp_seq )
{
        num_lost = recvd_esp_seq - expected_esp_seq;
}

for ( i = 0; i < num_lost && expected_seq != recvd_seq; i++ )
{
    if (expected_esp_seq part of in-sequence blocks DSTOPT)
    {
        loss = corruption;
    }
    else
    {
        loss = congestion;
    }
    expected_seq++;
}
```

Table 4.1: SPEP loss detection and distinction algorithm

numbers are the ones starting from the expected sequence number till the received sequence number. The expected sequence number is then searched in the option field of DSTOPT header. The option field of the DSTOPT header is implemented using the TCP sticky options described by Advanced Sockets API for IPv6 [41]. If the expected sequence lies within the range of any of the blocks of in-sequence numbers carried in DSTOPT header, then the loss is determined to be caused by corruption else it is concluded to be a congestive loss.

The loss recovery mechanism operates at the transport layer by using the loss information provided by the loss detection and distinction mechanism. The transfer of information from the network layer to the transport layer is explained next. In FreeBSD, the Internet protocol layer maintains a protocol control block, from now on referred to as INPCB, which among other things contains information about the connection end point identifiers and a pointer to the TCP Control Block (TCPCB)

[42]. The TCPCB maintains the entire state information required by each TCP connection. The loss detection and distinction mechanism stores the determined loss information in INPCB which is eventually copied into the TCPCB as the packet traverses up the protocol stack. TCPCB contains a pointer to the head of TCP reassembly queue. The TCP reassembly queue is a linked-list of out-of-order packets used to arrange the packets in sequence before delivering data to the application layer. The loss information is conveyed to respective element in the reassembly queue. The loss recovery mechanism works by sending ACKs to recover the lost packet by enabling the ELN bit in the header if the loss is determined to be by corruption.

The sender reacts to the ELN bit in the TCP header in accordance with the proposed ELN scheme in [29]. On receipt of an ACK with ELN bit set, the sender immediately retransmits the lost packet without invoking congestion control.

4.3 Behavior of SPEP under Different Conditions

This section describes the behavior of SPEP under conditions such as packet re-ordering and mobile handoff. It is essential to ensure the proper working of SPEP in the conditions mentioned above which are quite common.

4.3.1 Presence of Packet Reordering

Previous research [43] indicates that packet reordering is not a rare phenomenon in the Internet. In this section, we show that, although the SPEP approach relies on blocks of in-sequence packets for loss detection and distinction, the working of SPEP is not affected by packet reordering.

Fixed Host		Packet	Mobile Host	
DSTOPT	ESP	Reordering	DSTOPT	ESP
0 - 0	1	1	1 - 1	1
0 - 0	2	3	1 - 1, 3 - 3	3
0 - 0	3	4	1 - 1, 3 - 4	4
0 - 0	4	2	1 - 1, 3 - 4, 2 - 2	2
0 - 0	5	5	3 - 4, 2 - 2, 5 - 5	5

Table 4.2: SPEP behavior in presence of packet reordering

Table 4.2 shows a simple contrived example to explain the working of SPEP in a situation where there is packet reordering. Column 'ESP' on the left, shows the packets transmitted from fixed host as 1, 2, 3, 4, 5. Column 'Packet Reordering' shows the sequence after packet re-ordering in the network as 1, 3, 4, 2, 5. Column 'ESP' on the right shows the packets arriving at the base station in the sequence 1, 3, 4, 2, 5. The SPEP populates the blocks of in-sequence numbers in the DSTOPT header as shown in Column Mobile Host. The SPEP mobile node component behaves as follows. The SPEP loss detection and distinction mechanism receives packet 1 followed by packets 3 and 4. On receipt of packet 3, the SPEP determines that packet 2 is lost due to congestion and the loss recovery mechanism at the transport layer sends an acknowledgement to recover the corresponding lost

segment. This process is repeated on receipt of packet 4. The SPEP loss detection and distinction mechanism has an additional check to handle packet reordering. If the sequence number of a received packet is less than the expected sequence number, the loss detection and distinction mechanism is bye-passed and the packet is directly processed by the loss recovery mechanism. Arrival of a packet with sequence number less than the expected sequence number indicates that the packet was delayed in the network. However, the SPEP loss detection and distinction mechanism would have already accounted for the delayed packet considering it to be lost. Therefore on arrival of the delayed packet the SPEP does not invoke the lost detection and distinction mechanism again, instead allows the delayed packet to be bye-passed to loss recovery mechanism. The loss recovery mechanism processes the received packet to complete the sequence in the TCP reassembly queue and passes the completed sequence to application layer. The loss recovery mechanism then sends ACK containing the next expected sequence. The working of SPEP in presence of packet reordering is in accordance with the standard TCP mechanism.

4.3.2 SPEP Mobile Handoff Scenario

As a mobile node moves, it may get connected to different base stations. Therefore, if any state information relevant to the ongoing connection is stored in the base station, it has to be migrated to the new base station, when the base station moves. The SPEP base station component maintains minimal state information at the base station. However, migration of such state information to the new base station will

not be necessary because the SPEP station component at the new node can perform independently.

Fixed Host		Base Station	Mobile Host	
DSTOPT	ESP		DSTOPT	ESP
0 - 0	1	1-1	1 - 1	1
0 - 0	2	1-2	1 - 2	3
0 - 0	3	3-3	lost	

Table 4.3: SPEP before handoff operation

Fixed Host		Base Station	Mobile Host	
DSTOPT	ESP		DSTOPT	ESP
0 - 0	4	4	4 - 4	4
0 - 0	5	5	4 - 5	5

Table 4.4: SPEP after handoff operation

Tables 4.3 and 4.4 show a contrived example to explain the working of SPEP in a handoff scenario. Table 4.3 shows the packets being processed by the SPEP base station. Table 4.3 shows that Packets 1, 2, 3 are received by the base station and marked accordingly by the SPEP base station component. Table 4.4 shows the case when the mobile node moves to a new base station. The new base station can process the packets independently without any state information from the previous base station. The new base station functions as follows: (i) identifies the flow using the triplet < source IP address,flow label,destination IP address >. (ii) Determines that it is a new flow and has no information of the last received packet. Therefore populates the DSTOPT header with received packet which is '4' in this case. There is no change in the functionality of the SPEP mobile node component and it behaves as explained in Section 4.2.1.

4.4 Summary

This chapter describes the second issue addressed by this thesis viz., Combined issue of performance and security in a wired-cum-wireless scenario. This chapter is started by describing the need for security in today's applications and the limitations of existing performance enhancement schemes to co-exist with end-to-end security. This is followed by a survey of various approaches and their limitations. A novel approach called SPEP has been presented in this paper which ensures end-to-end security as well as enhances performance of TCP over wireless links. The design considerations, the implementation details and behavior of SPEP in various scenarios are described.

"If you want to make an apple pie from scratch, you must first create the universe."

- Carl Sagan

5

SPEP: Test Methodology and Performance Evalulation

In order to build and evaluate SPEP scheme, the first and foremost requirement was a test bed. Some obstacles mentioned in Section 5.4 had to be overcome and most of existing performance tools had to be enhanced to setup an IPSEC enabled IPv6 test bed for SPEP implementation and performance evaluation.

5.1 Test Configuration

A controlled network environment as shown in Figure 5.1 was set up to carry out the tests. A site local IPv6 address was assigned to each machine. The Sender and the Receiver machines were separated by 3 machines each of which was designated with

a specific functionality. IPSEC was configured to encrypt the traffic using transport mode ESP protocol between Sender and Receiver. SPEP mobile node component was installed on the Receiver. The SPEP base station component was installed on the machine named SPEP (base station). The ELN implementation was installed on the Sender. The corruption emulation used in Snoop [18] experiments was

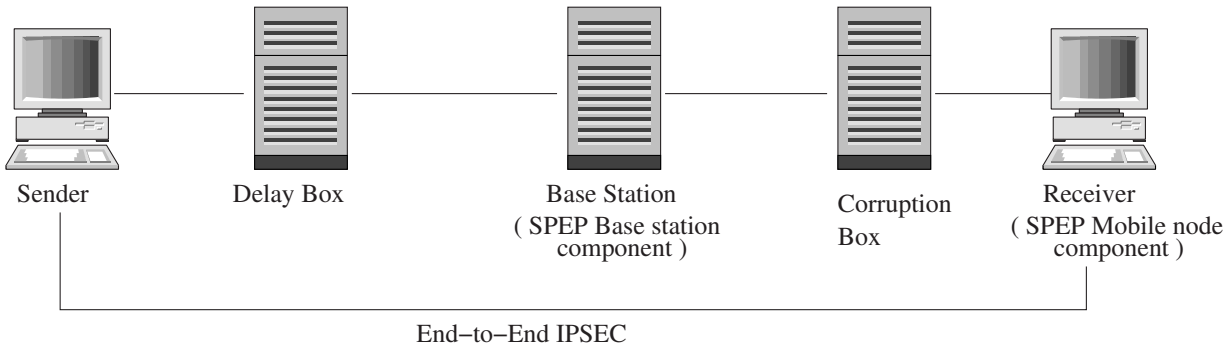


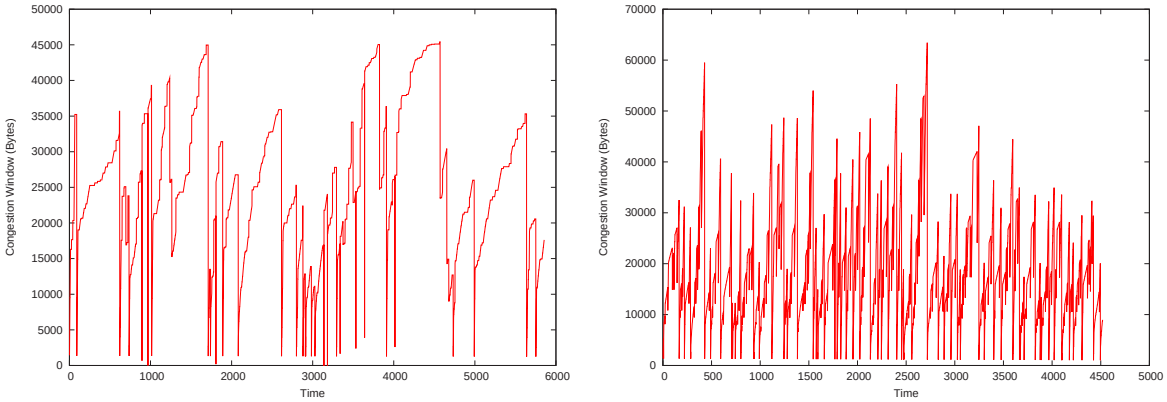
Figure 5.1: SPEP test configuration

enhanced to support IPv6-IPSEC and installed in Corruption Box. All the machines mentioned above were running FreeBSD 4.5. The rshaper [44] tool was enhanced to support IPv6-IPSEC and installed in the Delay Box to generate delay.The traffic was generated using iperf tool [45]. Iperf was enhanced by using TCP sticky option [41] to support DSTOPT header. TCPDUMP [46] with IPv6-IPSEC support was used for network monitoring.

5.2 Performance Evaluation

The test configuration described in the previous section is used to conduct the experiments. Tests were carried out with an intent to: (i) evaluate the performance

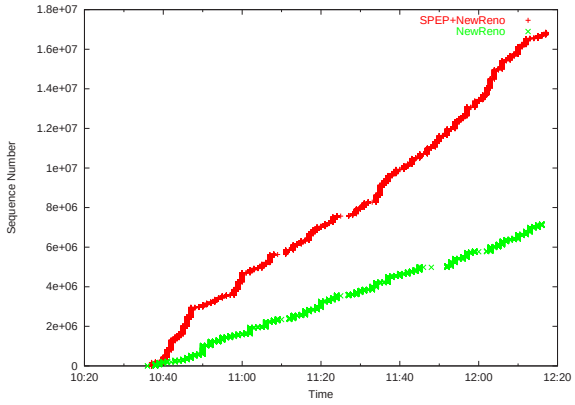
improvement achieved by SPEP scheme over traditional TCP New Reno for varying levels of corruption in the wireless link; (ii) analyze the achieved performance improvement; (iii) to estimate the overhead of SPEP scheme. All the test runs were conducted by making a bulk data transfer from sender to receiver for 100s. All the measurements were averaged across 10 test runs.



(a) Congestion window variation with SPEP (b) Congestion window variation without SPEP

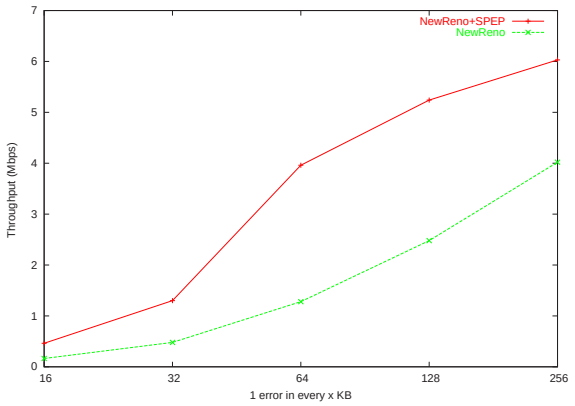
Figure 5.2: Congestion window variation for LAN scenario (1 error in every 32KB)

Tests were conducted by setting a 20ms delay in the Delay Box for LAN experiments and by setting a 200ms delay for WAN experiments. Figures 5.2 - 5.4 show the performance improvement achieved by SPEP scheme in a LAN scenario for different levels of corruption. The SPEP scheme distinguishes between congestion and corruption losses and informs the TCP sender to avoid invocation of congestion control for corruption losses. Figure 5.2, shows that the variation of congestion window with SPEP scheme is relatively less compared to the case without SPEP. The number of retransmission timeouts (indicated by the number of times the congestion window is reset to one segment) is much less with SPEP scheme. As can be seen in the time sequence graph in Figure 5.3 SPEP scheme is able to transmit



(a)

Figure 5.3: Time-Sequence graph LAN scenario (1 error in every 32KB)

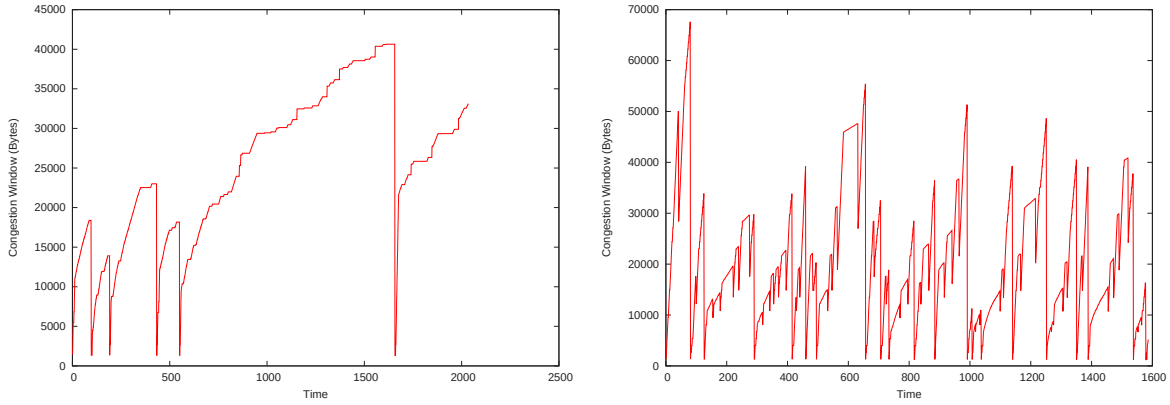


(a)

Figure 5.4: Throughput of New Reno with and without SPEP for LAN scenario

more packets in the given time, indicated by steeper slope.

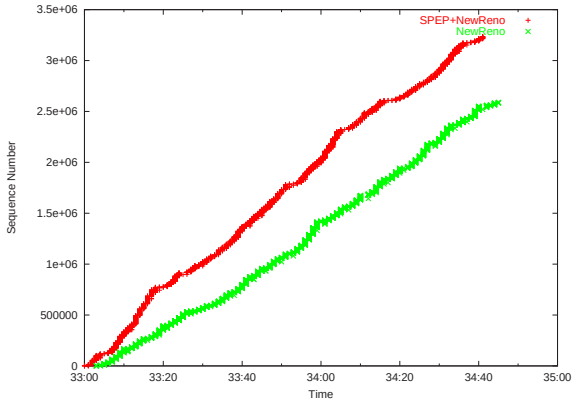
The performance improvement achieved by SPEP over standard TCP New Reno in a WAN scenario for different levels of corruption can be seen in Figures 5.5 - 5.7. The reduced number of retransmission timeouts and relatively less variation in congestion window of SPEP scheme as seen in Figure 5.5 explains the reason for increased throughput. The time sequence graph in Figure 5.6 further justifies the above increase in throughput.



(a) Congestion window variation with SPEP (b) Congestion window variation without SPEP

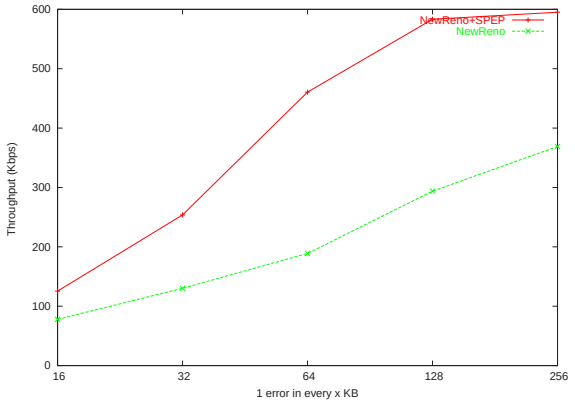
Figure 5.5: Congestion window variation for WAN scenario (1 error in every 32KB)

A comparison was made between the SPEP scheme and E2E-ELN scheme in an IPSEC environment, to estimate the communication and processing overhead introduced by SPEP scheme. Figure 5.8 shows that overhead introduced by the SPEP scheme is minimal. An implementation strategy for E2E-ELN is mentioned in [47]. However, in practice, it is not possible to realize an effective E2E-ELN scheme to co-exist with IPSEC. The decryption of a corrupted packet at the receiver garbles the entire packet. Hence, E2E-ELN scheme proves to be ineffective with application of security. We implemented a simple mechanism to emulate E2E-ELN scheme. We used test configuration shown in Figure 2 and ensured absence of any congestion between sender to base station. The Corruption Box was removed and the Receiver possessed the logic to drop 1 packet in say 'x' KB. The information of the packet dropped was used for loss recovery using ELN.



(a)

Figure 5.6: Time-Sequence graph WAN scenario (1 error in every 32KB)



(a)

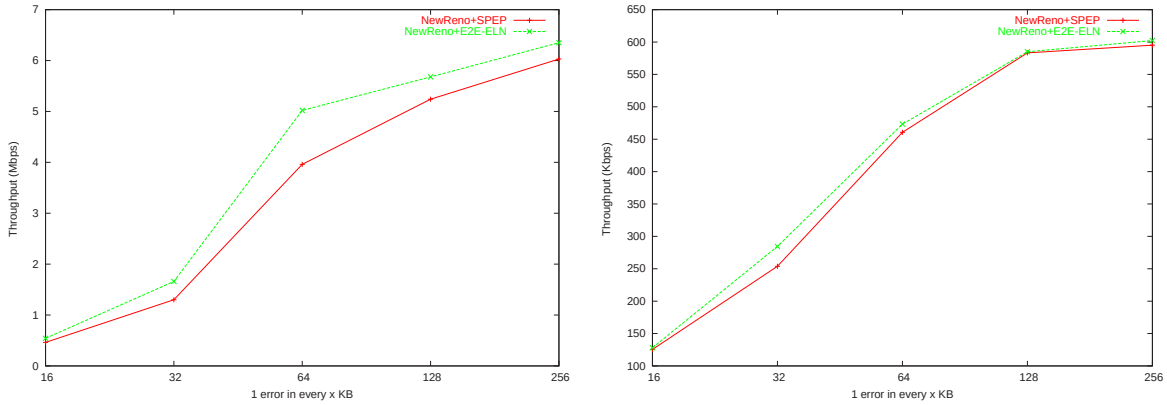
Figure 5.7: Throughput of New Reno with and without SPEP for WAN scenario

5.3 SPEP Approach: Merits

RFC 3135 [48] describes some of the implications of using PEP. SPEP addresses all the implications as detailed below.

The End-to-end Argument

End-to-end argument is considered to be one of the architectural principles in the design of the Internet, the violation of which is unacceptable. The SPEP approach



(a) LAN Scenario (b) WAN Scenario

Figure 5.8: Throughput of SPEP with NewReno V/s SPEP

complies with the end-to-end argument:

(i) Security: The SPEP approach ensures end-to-end security along with providing end-to-end performance. All the security issues that arise by having intermediate node as a PEP [48] are dealt with.

(ii) Fate Sharing: Fate sharing is used to describe the amount of state information on behalf the connection present in the network and also whether such state is self-healing in an event of network failure. Fate sharing also mentions that the user should be provided with a choice of using PEP. The SPEP approach does not maintain any state information in the network. The SPEP components are activated only on the receipt of the particular option in DSTOPT header. Hence, the user application has the option to enable/disable this option.

(iii) End-to-end reliability: It is recommended that data should be acknowledged end-to-end to ensure end-to-end reliability. SPEP adopts the above said approach.

(iv) End-to-end Failure Diagnostics: End-to-end loss recovery is done in SPEP, thus circumventing all the concerns expressed in [48]

Asymmetric Routing

The SPEP approach operates by racking only data packets in the forward direction and does not need to track the ACK packets in the reverse direction. Hence, SPEP works even in cases where data and ACK packets traverse in different paths.

Handoff Performance

The SPEP approach does not maintain any state information in the intermediate node(s). The SPEP base station component only needs to keep track of blocks of in-sequence ESP sequence numbers, for a specific connection as explained in section 4.1. In the case of mobility, the new SPEP base station starts afresh and easily performs the above said functionality, without requirement for any additional state information.

Scalability

Most of the previous PEP implementations are implemented above the IP layer and are not scalable as they maintain a lot of state information. The SPEP approach is scalable, the SPEP base station component operates at the IP layer and the nature of processing is comparable to marking of a ingress router in Diffserv architecture.

Generic Network layer Proxy

SPEP can also serve as a generic proxy for all transport layer protocols be it TCP, UDP with TFRC or any other transport layer protocol. SPEP can easily be enhanced to support TFRC flows, thereby improving the performance of real time applications over Wireless.

5.4 Problems Encountered

- Non-availability of a stable version of IPSEC enabled IPv6 Linux kernel. There were some problems in using beta release of Linux FreeSWAN (IPSEC patch for IPv6). Hence KAME FreeBSD chosen for implementation.
- IPv6 Destination Option support was needed in the traffic generator. However, IPv6 Advanced Socket API does not support TCP sockets to send ancillary data. It was resolved using TCP Sticky Options suggested in RFC 2292.
- Most TCP performance tools like tcptrace, iperf, rshaper lack IPv6 and/or IPSEC support. Scenarios cannot be emulated and implementation cannot be tested without such tools. The issue was resolved by enhancing the tools to support IPv6/IPSEC.
- Problems were encountered in managing and establishing IPv6 site-local routing due to statically configured routes. It was resolved by using Zebra protocol and running OSPFv6 on all the nodes.

5.5 Summary

This chapter describes the test methodology adopted to evaluate the performance of SPEP. The results of the various test bed experiments are presented followed by enumerating the problems encountered during the implementation.

"A conclusion is the place where you got tired thinking."

- Martin Henry Fischer

6

Conclusion

In this chapter, the thesis is concluded with a summary of contributions followed by a review of thesis objectives and some directions for future work.

6.1 Summary

Many solutions have been proposed over the years to improve the performance of TCP. However, some approaches address the problem with a myopic view. Although such approaches resolve the issue at hand they are not applicable to other scenarios. In this thesis two specific issues have been examined viz., (i) Effect of algorithms that improve fairness of TCP congestion avoidance on the performance of slow links and long thin networks; (ii)Combined issues of end-to-end security and performance in a wired-cum-wireless scenario.

In the first part of the thesis, the limitations of fairness algorithms are identified and a detail study is conducted to examine their effects on connections traversing slow links and long thin networks. Simulation experiments were conducted using a topology used commonly by the research community for fairness related experiments. The simulation results show that the algorithms to improve fairness have adverse effects on connections traversing either slow links or LTN. Different methods were used to counter the performance degradation, namely (i) increasing the buffer size at the last hop router (ii) advertising a smaller receive window (iii) selectively disabling the policies. It has been argued that it is not appropriate to apply the fairness algorithms to connections that traverse slow links or LTNs. Test bed experiments were also conducted by setting up a controlled network environment and using live GSM network. The results of our test bed experiments concur with the simulation results.

In the second part of the thesis, it is discussed that most of the existing solutions to improve the performance of TCP in a wired-cum-wireless scenario, fail to work when end-to-end security schemes are applied. Extensive survey was conducted and it was identified that the IP security and TCP performance have so far been treated in a mutually exclusive manner. An innovative mechanism, Secure Performance Enhancing Proxy (SPEP) was proposed, to address the seemingly arduous problem of enhancing TCP performance over wireless networks, preserving end-to-end TCP semantics as well as ensuring end-to-end security. The design of SPEP leverages on the features of IPv6 to provide security as well as performance enhancement for TCP connections in a wired-cum-wireless environment. The proposed SPEP

scheme decouples error detection and error distinction mechanism from error recovery mechanism which not only facilitates in performance improvement but also offers multifarious advantages discussed in the paper. The proposed scheme was implemented in FreeBSD 4.5 and experiments were conducted in a controlled test bed setup. The results show improved TCP performance in a secured environment with introduction of minimal overhead.

6.2 Review of thesis objectives

In the first chapter, the objectives of thesis and the related research work was laid down as :

- Study the existing solutions to improve the performance of TCP.
- Identify the issues with the proposed solutions and zero in on specific issues for further study.
- Investigate into the issues concerning the applicability and/or limitations of existing schemes in the context of the last mile wireless scenario.
- Provide efficient solutions to overcome the limitations.
- Simulation / test bed experiments to substantiate and verify the performance improvement.

Extensive survey was done to explore the various schemes proposed to improve the performance of TCP. Two specific issues were identified for the study namely (i) Effect of algorithms that improve fairness of TCP congestion avoidance on the

performance of slow links and long thin networks; (ii)Combined issues of end-to-end security and performance in a wired-cum-wireless scenario.

Extensive simulation and test bed experiments were conducted to bring to light the limitations of the existing schemes to improve fairness algorithms.Various methods were suggested to counter the performance degradation and suggested methods were evaluated by simulations and test bed experiments.

The literature survey identified that most of the solutions previously proposed to address the problem of enhancing TCP performance in a hybrid wired-wireless network are designed oblivious of the security considerations and violate end-to-end semantics.A novel approach called SPEP has been presented which ensures end-to-end security as well as enhances performance of TCP over wireless links. The multifarious advantages obtained by using SPEP approach was also discussed. The SPEP protocol was implemented in FreeBSD 4.5. Performance evaluation was done in a controlled network environment and results show remarkable improvement in performance with minimal overhead.

6.3 Future Work

The SPEP approach described in this paper, offers a unique solution at the network layer which can be readily extended to support applications using TFRC over wireless. SPEP can be designed to be a generic framework to support applications using TFRC and also other transport layer protocols. SPEP approach uses cumulative

acknowledgements for loss recovery, it could be extended to support SACK. SACK enabled SPEP scheme would be capable of recovering from multiple losses in a window and hence result in improved performance. SPEP scheme can be evaluated in a real network for LAN and WAN scenarios.

The current SPEP scheme is designed to use the ELN mechanism to inform the sender about the nature of loss and prevent the invocation of congestion control if the loss is due to corruption. This would require a modification of TCP stack at the sender. The sender side modification can be overcome when SPEP is designed to use delayed duplicate acknowledgement scheme.

Bibliography

[1] C. M. Cordeiro, H. Gossain, R. L. Ashok, and D. P. Agarwal, "The Last Mile: Wireless Technologies for Broadband and Home Networks," Technical Report, Center for Distributed and Mobile Computing, ECECS, University of Cincinnati, January 1997.

[2] S. M. Cherry, "The Wireless Last Mile," *Spectrum, IEEE*, vol. 40(9), pp. 18–22, 2003.

[3] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya, "End-to-End Performance Implications of Slow links," RFC 3150, July 2001.

[4] G. Montenegro, S. Dawkins, M. Kojo, V. Magret, and N. Vaidya, "Long Thin Networks," RFC 2757, January 2000.

[5] V. Jacobson, "Congestion Avoidance and Control," *Computer Communication Review*, vol. 18(3), pp. 314–329, 1998.

[6] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms," RFC 2001, January 1997.

[7] T. Henderson, E. Sahouria, S. McCanne, and R. Katz, "On Improving the Fairness of TCP Congestion Avoidance," in *Proceedings of IEEE Globecom*, 1998.

[8] S. Floyd and V. Jacobson, "On Traffic phase effects in Packet Switched Gateways," *Journal of Internetworking Practice and Experience*, vol. 3(3), pp. 115–156, 1992.

[9] Sally Floyd, "Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: One way Traffic," *ACM Computer Communications Review*, vol. 21, 1991.

[10] H. Benaboud and N. Mikou, "CANIT : A New Algorithm to improve the fairness of TCP Congestion Avoidance," in *Proceedings of 6th IEEE Symposium on Computers and Communication*, 2001.

[11] W.R. Stevens, "TCP/IP Illustrated,The Protocols," vol. 1. Addison-Wesley Longman, N.Delhi, India, 2001.

[12] E. Ayanoglu, S. Paul, T.F LaPorta, K.K Sabnani, and R.D Gitlin, "AIR-MAIL: A Link-Layer Protocol for Wireless Networks," *ACM/Baltzer Wireless Networks*, vol. 1(1), pp. 47–60, 1995.

[13] A. Bakre and B.R. Badrinath, "I-TCP : Indirect TCP for Mobile Hosts," in *Proceedings of 15th International Conference on Distributed Computing Systems*, May 1995.

[14] R. Yavatkar and N. Bhagawat, "Improving end-to-end Performance of TCP over Mobile Internetworks," in *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, December 1994.

[15] T. Goff, J. Moronski, D. S. Phatak, and V. Gupta, "Freeze-TCP: A True End-To-End TCP Enhancement Mechanism for Mobile Environments," in *IEEE Conference on Computer Communications*, June 2000.

[16] R.K. Balan, B.P. Lee, K.R.R. Kumar, L. Jacob, W.K.G. Seah, and A.L. Ananda, "TCP HACK: TCP Header Checksum Option to Improve Performance over Lossy Links," in *Proceedings of IEEE INFOCOMM*, April 2001.

[17] M. Mathis, J. Mahdavi, S. Floyd, and Romanow, "TCP Selective Acknowledgement Options," RFC 2018, April 1996.

[18] E.Amir R. H. Katz H. Balakrishnan, S. Seshan, "Improving TCP/IP Performance over Wireless Networks," in *Proceedings of ACM Mobicom*, November 1995.

[19] T. Shepard and C. Patridge, "When TCP starts up with Four packets into only 3 buffers," RFC 2416, September 1998.

[20] P. Sarolahti, "Performance evaluation of TCP Enhancements Over Slow Wireless links," in *Proceedings of Finnish Data Processing Week*, Petrozavodsk, Russia, 2001.

[21] A. Gurtov, "TCP Performance in presence of Congestion and Corruption losses," M.S. thesis, Department of Computer Science, University of Helsinki, 2001.

[22] "Network Simulator (NS)," Available at http://www.isi.edu/nsnam/ns.

[23] T. Henderson and R. Katz, "TCP Over Satellite Channels," *Technical Report* , 1999, University of California, Berkley.

[24] H. Balakrishnan, V. N. Padmanabhan, and R. H. Katz, "The Effects of Asymmetry on TCP Performance," *ACM Mobile Networks and Applications (MONET)*, vol. 4(3), 1999.

[25] "Dummynet," Available at http://info.iet.unipi.it/ luigi.

[26] "Nokia D211," Available at http://www.nokia.com/nokia/0,4879,1449,00.html.

[27] K. Brown and S. Singh, "M-TCP: TCP for Mobile Cellular Networks," *ACM Computer Communications Review (CCR)*, vol. 27, pp. 5, 1997.

[28] K. Ratnam and I. Matra, "WTCP: An Efficient Mechanism for Improving TCP Performance over Wireless Links," in *IEEE Symposium on Computers and Communications*, June 1998.

[29] H. Balakrishnan and R.H Katz, "Explicit Loss Notification and Wireless Web Performance," in *Proceedings of IEEE Globecom Internet Mini-Conference*, November 1998.

[30] S. Bellovin and M. Blaze, "Cryptographic Modes of Operation for the Internet," in *Second NIST Workshop on Modes of Operation*, 2001.

[31] M. Mehta and N.H Vaidya, "Delayed Duplicate Acknowledgements: A proposal to Improve Performance of TCP on wireless links," Technical Report,Texas A&M University, December 1997.

[32] S. Biaz and N. Vaidya, "Distinguishing congestion losses from wireless transmission losses: A negative result," in *Proceedings of International Conference on Computer Communications and Networks*, October 1998.

[33] N. Assaf, J. Luo, M. Dillinger, and L. Menendez, "Inter-working of IP Security and Performance Enhancing Proxies," *IEEE Communications Magazine*, May 2002.

[34] T. Dierks and C. Allen, "The TLS Protocol Version 1.0," RFC 2246, January 1999.

[35] Y. Zhang and B. Singh, "A Multi-Layer IPSec Protocol," in *Proceedings of 9th Usenix Security Symposium*, August 2000.

[36] S. Bellovin, "Probable Plaintext Cryptanalysis of the IPSecurity Protocols," in *Proceedings of the Symposium on Network and Distributed System Security*, February 1997.

[37] S. Deering and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460, December 1998.

[38] N. Ferguson and B. Schneier, "A Cryptographic Evaluation of IPsec," Available at http://citeseer.nj.nec.com/ferguson00cryptographic.html.

[39] S. Kent and R. Atkinson, "IP Authentication Header," RFC 2402, November 1998.

[40] S. Kent and R. Atkinson, "IP Encapsulating Security Payload (ESP)," RFC 2406, November 1998.

[41] W. Stevens and M. Thomas, "Advanced Sockets API for IPv6," RFC 2292, February 1998.

[42] G. Wright and W.R. Stevens, "TCP/IP Illustrated,The Implementation," vol. 2. Addisson-Wesley, N.Delhi, India, 1998.

[43] J.C.R Bennett, C. Partridge, and N. Shectman, "Packet reordering is not pathological network behavior," *IEEE/ACM Transactions on Networking (TON)*, vol. 7(8), pp. 789–798, 1999.

[44] "Rshaper," Available at : http://www.ar.linux.it/pub/rshaper.

[45] "Iperf," Available at : http://dast.nlanr.net/Projects/Iperf.

[46] "Tcpdump," Available at : http://www.tcpdump.org.

[47] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," *IEEE/ ACM Transactions on Networking*, 1996.

[48] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby, "Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations," RFC 3135, June 2001.

A

Appendix I

A.1 Papers published related to thesis

- V. Obanaik, L. Jacob, A.L. Ananda, "Effect of Algorithms that Improve Fairness of TCP Congestion Avoidance on Performance of Slow Links and Long Thin Networks",In *Proceedings of 11th International Conference on Computer Communications and Networks*, October 2002.
- G. Poduval, V. Obanaik, A.L.Ananda, "Impact of Fairness Policies on Slow Links and Long Thin Networks", In *TENCON IEEE Region 10 Conference on Networking*, October 2003.
- V. Obanaik, L. Jacob, A.L Ananda, "SPEP: A Secure and Efficient Scheme for Bulk Data Transfer over Wireless Networks", accepted for publication in *IEEE Wireless Communications and Networking Conference*, March 2004.

TCP	Transmission Control Protocol
RTT	Round Trip Time
ACK	Acknowledgement
LTN	Long Thin Networks
IBK	Increase-by-K
CR	Constant Rate
CANIT	Congestion Avoidance in Normalized Interval of Time
PEP	Performance Enhancing Proxy
SPEP	Secure Performance Enhancing Proxy
FEC	Forward Error Correction
ARQ	Automatic Repeat Request
ISP	Internet Service Provider
MTU	Maximum Transmission Unit
IPSEC	IP Security
AH	Authentication Header
ESP	Encapsulating Security payload
SSL	Secure Sockets Layer
TLS	Transport Layer Security