DETERMINATION OF NETWORK DELAY DISTRIBUTION OVER THE INTERNET

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

MEHMET KARAKAŞ

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2003

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Mübeccel DEMİREKLER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Faruk Rüyal ERGÜL
Supervisor

Examining Committee Members

Prof. Dr. Mete SEVERCAN _____

Prof. Dr. Faruk Rüyal ERGÜL _____

Prof. Dr. Yalçın TANIK _____

Prof. Dr Hasan GÜRAN _____

Serkan SEVİM _____

ABSTRACT

DETERMINATION OF NETWORK DELAY DISTRIBUTION OVER THE INTERNET

KARAKAŞ, Mehmet

M.S., Department of Electrical and Electronics Engineering

Supervisor : Prof. Dr. Faruk Rüyal ERGÜL

December 2003, 93 pages

The rapid growth of the Internet and the proliferation of its new applications pose a serious challenge in network performance management and monitoring. The current Internet has no mechanism for providing feedback on network congestion to the end-systems at the IP layer. For applications and their end hosts, end-to-end measurements may be the only way of measuring network performance.

Understanding the packet delay and loss behavior of the Internet is important for proper design of network algorithms such as routing and flow control algorithms, for the dimensioning of buffers and link capacity, and for choosing parameters in simulation and analytic studies.

In this thesis, round trip time (RTT), one-way network delay and packet loss in the Internet are measured at different times of the day, using a Voice over IP (VoIP) device. The effect of clock skew on one-way network delay measurements is eliminated by a Linear Programming algorithm, implemented in MATLAB. Distributions of one-way network delay and RTT in the Internet are determined. It is observed that delay distribution has a gamma-like shape with heavy tail. It is tried to model delay distribution with gamma, lognormal and Weibull distributions. It is observed that most of the packet losses in the Internet are single packet losses. The effect of firewall on delay measurements is also observed.

Keywords: Internet, Voice over IP (VoIP), end-to-end measurement, one-way network delay, RTT, packet loss, clock skew

ÖZ

İNTERNET'TE AĞ GECİKMESİ DAĞILIMININ BELİRLENMESİ

KARAKAŞ, Mehmet

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi : Prof. Dr. Faruk Rüyal ERGÜL

Aralık 2003, 93 sayfa

İnternet'in hızlı büyümesi ve yeni uygulamalarının çoğalması ağ performansının yönetimi ve izlenmesinde ciddi zorluklar ortaya çıkarmaktadır. Şu anki İnternet'in IP seviyedeki uç-sistemlere, ağdaki tıkanıklıklarla ilgili geribesleme verecek bir yapısı yoktur. Uygulamalar ve sunucuları için ağ performansını ölçmenin tek yolu uçtan-uca ölçümler olabilir.

İnternet'in paket gecikmesi ve paket kaybı davranışının anlaşılması akış kontrolü algoritmaları gibi ağ algoritmalarının uygun olarak tasarlanmasında, tampon boyutlarının ve hat kapasitesinin belirlenmesinde, simülasyon ve analitik çalışmalarda parametre seçiminde önemlidir.

Bu tez çalışmasında, İnternet'teki dairesel döngü zamanı (DDZ), tek yönlü gecikme ve paket kaybı günün değişik zamanlarında, İnternet Protokolü Üzerinden Ses (İPÜS) cihazı kullanılarak ölçülmüştür. Tek yönlü gecikme ölçümlerimizdeki saat eğriliğinin etkisi MATLAB'da gerçeklenen Doğrusal Programlama algoritması ile yok edilmiştir. İnternet'teki DDZ ve tek yönlü gecikme dağılımı belirlenmiştir. Gecikme dağılımının ağır kuyruklu gamma benzeri bir şekilde olduğu gözlemlenmiştir. Gecikme dağılımının gamma, lognormal ve Weibull dağılımları ile modellenmesine çalışılmıştır. İnternet'teki paket kayıplarının genelde tek paketlik kayıplar şeklinde olduğu gözlemlenmiştir. Aynı zamanda, ateş duvarının gecikmeler üzerindeki etkisi de gözlemlenmiştir.

Anahtar Kelimeler : İnternet, İnternet Protokolü Üzerinden Ses (İPÜS), uçtan-uca ölçüm, tek yönlü gecikme, DDZ, paket kaybı, saat egriliği

To My Family

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Prof. Dr. Faruk Rüyal Ergül for his supervision, valuable guidance and helpful suggestions.

I would like to express my sincere appreciation to Turgut Çelikadam, for providing us VoIP device for our measurements. He also helped us for adding extra functions for delay measurements to VoIP device. And also I want to thank to Hasan Çitçi and my friends in ASELSAN Inc. for their valuable friendship, help and support. I want to thank to Şanser Şirin from Hacettepe University and Oktay Koç from Middle East Technical University for their help and support during measurements. And also I want to thank to Mehmet Celep, Alparslan Gözel and Halil İbrahim Seyrek from Gebze Institute of Technology for their help and support during measurements. I am also grateful to ASELSAN Inc. for facilities provided for the completion of this thesis.

I am grateful to my wife, Esin, because she always supported me and cheered me up when I needed.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

CHAPTER 1

INTRODUCTION

Over the last decade Internet has grown by orders of magnitude in size. The Internet is now widely deployed and the users can easily get the global accessibility from their home terminals. One of the main reasons for the prevalence of the Internet is in its routing mechanism. Routing of the Internet has two key features; flexibility and scalability. First, the Internet provides the dynamic routing by exchanging the routing information among routers. Then, if a network link becomes down because of some troubles, an alternative route will be prepared automatically. Second, the packet processing at the router is simple (e.g. FIFO) to reduce the overhead of packet forwarding at the router.

The Internet serves a best effort service, which means basic connectivity with no guarantees. As available bandwidth increased, new applications such as Internet telephony, audio and video streaming services, video-on-demand, and distributed interactive games have proliferated. These new applications have diverse quality-of-service (QoS) requirements that are significantly different from traditional best-effort service. QoS can be defined as an idea that transmission rates, error rates and other characteristics can be measured, improved, and, to some extent, guaranteed in some advance.

High-quality video applications, such as, remote medical diagnosis and video-on-demand applications, demand reliable and timely delivery of high bandwidth data

and requires QoS guarantees from network. On the other hand, a majority of multimedia applications including Internet telephony, video conferencing, and web TV, do not need in-order, reliable delivery of packets, and can tolerate a small fraction of packets that are either lost or highly delayed, while still maintaining reasonably good quality. These applications employ end-to-end control that adapts to the changing dynamics of the network and can deliver the acceptable quality to users. QoS can be characterized with a number of parameters. The most important ones are packet loss, end-to-end delay and jitter.

Packet loss is an important parameter of the network QoS. Typically more than 5% lost packets will annoy users at VoIP applications [1]. Packet loss usually occurs when there is congestion on the packets path, causing router buffers to overflow. It is heavily influenced by the route stability of the network, efficient queue management in the routers, and proper use of congestion control. Two schemes are used to deal with packet loss;

- Automatic Repeat Request (ARQ) is based on the retransmission of packets that are not received at the destination. This is generally not used by live audio applications because the retransmission increases the end-to-end delay.

- Forward Error Correction (FEC) is based on the transmission of redundant information along with the original information so that the lost original data can be recovered from the redundant information.

End-to-end delay is the time that it takes a packet to make its way through a network from a source to a receiver. Big delay makes conversation unnatural and annoying. The International Telecommunications Union (ITU) standard G.114 states that a one-way-delay less than 150 ms is acceptable for good quality phone calls [2]. The total one-way delay is the sum of a packet assembly at the source, a

network delay, and a receiver delay. The delay at the source is for balancing network efficiency and the response needed in an interactive system. The network delay is beyond the control of the end user. The delay at the receiver is inserted to compensate for network delay. Network delay, which is one of the factors we are interested in, is mainly composed of propagation, serialization, switching/routing and queuing delay.

- Serialization delay is the time it takes to serialize the digital data onto the physical links of the interconnecting equipment. That is how long it takes to put the bits on the wire. The faster the media, the lower the delay. This delay is unavoidable, but can be reduced by keeping the number of intervening links small and using high bandwidth interfaces.

- Propagation delay is the time it takes the signal to travel the physical distance from end-to end. The propagation delay is determined by the travel time of an electromagnetic wave through the physical channel of the communication path and is independent of actual traffic on the link. There is always propagation delay, but it is usually small. It only becomes an issue when packets travel a long physical distance, for example over satellite links.

- Switching/routing delay is the time the router takes to switch the packet. This time is needed to analyze the packet header, check the routing table, and route the packet to the output port. This delay depends on the architecture of the route engine and the size of the routing table. New IP switches can significantly speed up the routing process by making routing decisions and forwarding the traffic via hardware as opposed to software processing.

- Queuing delay, which is a large source of latency, is the amount of time that a packet remains buffered in a network element while it waits transmission.

Network traffic loads result in variable queuing delays. For an unloaded network this delay is negligible. For a network that is heavily congested, it is usually the main delay component. The variable queuing delay serves a very important role in network and application design. Continuous-media applications such as audio and video need to absorb the delay jitter perceived at the receiver for smooth play-out of the original stream [3, 4]. Determining the correct amount of buffering, and reconstructing the original timing is crucial to the performance of continuous-media applications. The variable queuing delay is also useful in monitoring network performance at the edges of the network; the transmission and propagation delays are fixed per route, and do not convey any information about the dynamic changes inside the network when packets follow a fixed route [5].

Jitter is the statistical variance of packet interarrival time, measured in milliseconds. It is defined to be the mean deviation (smoothed absolute value) of the packet spacing change between the sender and the receiver [6]. Packets transmitted at equal intervals from the sender arrive at the receiver at irregular intervals. Jitter is introduced due to the internal operations of the components in the network. Queuing and buffering of the data in the network, packet rerouting, packet loss, network multiplexing and other such factors can cause jitter. For high-quality voice, the average inter-arrival time at the receiver should be nearly equal to the inter-packet gaps at the transmitter. Jitter buffers are used to compensate for the delay variance. They hold incoming packets for a specified amount of time to allow the slowest packets to arrive before they are delivered to the end user in a more constant stream. A jitter buffer introduces additional delay. The size of the jitter buffer can be optimized to minimize the delay. However, a too small jitter buffer decreases the delay but increases the packet loss, resulting in bad quality. A too large jitter buffer does not degrade the voice quality, but increases the delay. There are two approaches to select the jitter buffer size. Fixed approach assumes that the range of delay is predictable and uses a static buffer size. Adaptive approaches, which are common in the Internet, measure immediate jitter and use that to dynamically adjust

the buffer size. Sharp variations in jitter values lead to a significant increase in packet loss.

The dynamic routing mechanism is important for IP (Internet Protocol) routing, but it is not always helpful for the end users. From the users' point of view, the packet transmission delay is an important performance metric since it directly affects the end-to-end quality. One example can be the real-time application using RTP (Real-time Transport Protocol) [6]; a popular protocol for real-time applications in recent years. RTP uses RTCP (Real-time Control Protocol) to control the transmission rate. In RTCP, the sender maintains the transmission delay of packets based on Round Trip Time (RTT) values to control the packet transmission rate. To keep the preferable performance in RTP-based applications, an accurate estimation of the packet transmission delay is essential. However, the dynamic routing of the Internet makes it impossible for the end-users to select the appropriate route for satisfying the users' quality of service (QoS). It is difficult even to predict the transmission delay of the packet, since a simple packet processing at routers provides a variable delay. Furthermore, in real-time applications (e.g., voice communications), it is desirable to separately measure transmission delays of both downstream (sender to receiver) and upstream (receiver to sender) routes because the Internet routes are often asymmetric due to its dynamic routing mechanism [7]. Even if downstream and upstream routes are on the same path, they may have radically different transmission delays due to time-fluctuating queuing delays at the routers. From these reasons, it is necessary to investigate not only the characteristic of RTT but also that of one-way transmission delays.

When all of the packets go through the same route to the receiver, they have the same propagation delay, and, if they have the same size, the serialization delay also is the same. Even if the packets go through the same route, and are of the same size, the packets experience different levels of queuing inside the network. This is what causes the variability in the end-to-end delay.

It is well known that the users of real time voice services are sensitive and susceptible to audio quality. If the quality deteriorates below an acceptable level or is too variable, users often abandon their calls and retry later. Since the Internet is increasingly being used to carry real time voice traffic, the quality provided has become, and will remain an important issue. As we have mentioned above paragraphs, Internet is a packet switched network, which gives best effort service. The Internet does not guarantee QoS, so the QoS parameters such as delay, packet loss and jitter must be investigated. The knowledge of the delay distribution along paths in the Internet allows us to verify whether QoS requirements of VoIP can be met. This information can be used to improve the performance of network parts that fail to offer acceptable delay bounds. From more academic point of view, these end-to-end delay measurements may give information about topology and traffic pattern of the Internet.

One-way network delay is important in voice communication, particularly if it is not equal in each direction. The accuracy of such measurements depends on whether the clocks involved in measurement are synchronized. Measuring the one-way delay of network connections without the use of synchronized clocks is a non-trivial task. Many methods rely on RTT measurements and halve the result, hence estimating the one-way delay. In this thesis, we did not use this method and attempted to eliminate the effects of unsynchronized receiver and transmitter clocks in order to obtain correct delay values.

In this thesis, delay distribution in the Internet is determined. Round Trip Time (RTT) and one-way network delay on two paths are measured. The first measurement was on the path Middle East Technical University (METU)-Hacettepe University (HU) on 28-29 June 2003. The second measurement was on the path Middle East Technical University-Gebze Institute of Technology (GIT) on 27 October to 14 November 2003. The effect of clock skew on one-way delay measurements is eliminated with Moon's linear programming algorithm by MATLAB [5]. Distributions of RTT and one-way network delay are analyzed using

statistics toolbox of MATLAB. It is attempted to find out if observed distributions fit to gamma, lognormal or Weibull distributions. In order to do this, maximum likelihood estimates of the parameters of these distributions are computed. Then, Kolmogorov-Smirnov goodness-of-fit test is applied to our observations and estimated distributions. None of the estimated distributions could pass the Kolmogorov-Smirnov goodness-of-fit test. Closeness of the estimated distributions and observed distributions is compared by the results of the Kolmogorov-Smirnov goodness-of-fit test.

During our measurements we have encountered the effect of firewall used in METU. A firewall is simply a program or hardware device that filters the information coming through the Internet connection into your private network or computer system. If an incoming packet of information is flagged by the filters, it is not allowed through. Firewalls are customizable which means that you can add or remove filters based on several conditions. Firewalls can also achieve virus control to the incoming packets. These filtering and virus control processes at the firewall can change the delay distribution and loss rate dramatically. HU and GIT do not use a firewall, but METU uses firewalls.

We have measured one-way network delay, RTT distribution and packet loss rate at the path METU-HU. We have observed high delay values and loss rates although HU and METU are not located apart. We have investigated the firewall rules of METU. VoIP devices used in our measurements communicate on port 1024. We have provided necessary rule changes at the firewall of METU so, incoming packets on port 1024 are not processed by the firewall of METU. Then, we have measured one-way delay and RTT on the path GIT-METU and observed that our results are similar with the results of Test Traffic Measurements (TTM) of Réseaux IP Européens Network Coordination Center (RIPE NCC). The results of this work can be used in developing play-out buffering algorithms and the measurements taken during this thesis can be used as input data for testing play-out buffering algorithms.

Organization of the thesis is as follows. In Chapter 2, previous work about measurements in the Internet and an ongoing world-wide measurement project is introduced. In Chapter 3, architecture of the measurement tool used in our measurements is introduced. In Chapter 4, measurement setup, measurement methods and implemented skew elimination algorithm are described. In chapter 5, data analysis method and obtained results are explained. The thesis ends in Chapter 6 with a summary and a discussion about the further scope for this work.

CHAPTER 2

RELATED WORK

2.1 Introduction

The rapid growth of the Internet and the proliferation of its new applications pose a serious challenge in network performance management and monitoring. As more users started to use new multimedia services over the Internet, monitoring network dynamics, and detecting any anomalies or breakdowns inside the network as soon as they appear become a more important issue.

The Simple Network Management Protocol (SNMP) was developed to provide a coherent framework in network management of the Internet. Using SNMP, network management stations send a request to end hosts and routers for variables defined by Management Information Base (MIB), and gather information about interfaces, routing tables, and protocol states from the replies. HP's OpenView, Desktalk Systems' TRENDsnmp+, and Castle Rock Computing's SNMPc are all SNMP-based network monitoring tools.

There are other tools that are available to network administrators; these monitors include OC3MON and OC12MON developed by MCI. They collect traffic data from high-volume trunks of OC3 and OC12 speed, (which correspond to 155.52Mb/s and 622Mb/s, respectively), and provide flow-based analysis of the

collected data. The detailed flow-based analysis allows us to identify the type of traffic and the source and destination of the traffic.

Cisco's Internetworking Operating System (IOS) has a similar passive monitoring feature called NetFlow. NetFlow allows the router to capture a number of traffic statistics, including a packet's source and destination IP addresses and port numbers, and protocol type. The collected data is then exported to another platform where the data can be used in network analysis, planning, management, accounting, billing, and data mining.

Unlike network administrators and managers, most end users and their applications do not have access to the inside of the network. The current Internet has no mechanism for providing feedback on network congestion to the end-systems at the IP layer. For applications and their end hosts, end-to-end measurements may be the only way of measuring network performance, especially when there is no provision inside the network to give information about the current status of the network to users without access to the routers. By measuring the delay, jitter and loss of the incoming data stream at the receiver, we can provide some indication on how suitable the network is for real time applications, like real time voice communication. Therefore the quality of VoIP sessions can be quantified by network delay, packet loss and packet jitter. These three quantities are the major contributors to the perceived quality as far as the network is concerned.

There are two modes of network performance measurement: active and passive. Active measurement tools generate packets and inject them to the network for the purpose of performance evaluation. All of the tools listed above are passive, since they do not generate traffic, but merely monitor the traffic on high-capacity trunks and routers.

Users at the edges of the network can learn about the conditions inside the network by actively generating packets at one end and observing the outcome at the other end of the network. Most end-to-end measurement-based tools fall into this category of active measurement, so does our tool. We have placed two VoIP devices at the end terminals and established voice communication in order to measure the one-way delay. During one-way delay measurements, voice packets are transmitted with 16 ms period and their size are 135 byte. While measuring RTT, we did not establish voice communication between two VoIP devices, instead, we have generated constant packets. Size and transmission period of these packets are adjustable, so we could measure the RTT values with different packet size and period.

Packet delay and packet loss are two fundamental end-to-end network characteristics that represent quality of service. The magnitude and variation of delay affect the interactivity of applications; the loss rate and the distribution of losses affect the loss recovery mechanism, and long-term throughput of applications.

2.2 Previous Work

There have been several studies undertaken to demonstrate the performance of the Internet, focusing on metrics such as loss, delay and jitter.

Paxson [8, 9] used TCP to determine network performance between 35 different pairs of hosts. His study revealed substantial variation in most metrics. He observed variations based on time of day, geography, and year. He observed loss probabilities that ranged between 0% and 65%. Paxson also found wide variability in the correlation of losses. From the set of traces collected during November to December of 1995, he examined the distribution of outages, which are the duration of time over which there is complete packet loss. He observed that 10% of the outages were

less than a few milliseconds, while another 10% were more than a few seconds. Similar variability is observed for delays.

Mukherjee [10] found that end-to-end one-way packet delays were well modeled using a shifted gamma distribution, but the parameters of the distribution depended on the path and time of day.

The study by Bolot [11] focused on a single link from INRIA in France to the University of Maryland in the U.S. They found that there was substantial correlation in delays, and demonstrated that this was due to their rapid probe packets piling up behind a large packet in a congested buffer.

Sanghi [12] used UDP to determine the connection properties between a number of hosts. They found that losses generally occurred one at a time.

In 1997, Maxemchuk and Lo measured quality of intra state, cross country, and international Internet links [13]. They defined the quality of a connection as “the fraction of time that the signal is received without distortion for intervals of time that are long enough to convey active speech segments”, and the measurements were based on UDP. The conclusions were that Internet was capable of carrying voice although there were problems; the quality could be improved by the proper combination of packet restoration and receiver delay. They had two important observations of how length of the connection and time of the day affected the quality of a connection; 1-) They have noted that the quality of international connections was not as good as cross country connections, which was not as good as the quality of intrastate connections, i.e., longer connections give worse quality. 2-) The time of day affected the quality very much. Quality was much worse during busy hours.

One of the early works of delay measurement was done by Bert Dempsey, Matthew Lucas and Alfred Weaver in 1994 at the University of Virginia, in which they studied round-trip delay of the University's campus network [14]. They used UDP to send a stream of datagrams to a remote machine in the network. Each datagram had a timestamp, used for calculating the round trip time, and a sequence number for detecting packet loss. On receiving of the datagram, the remote machine immediately transmitted a datagram of the same size and the same sequence number. Delay was calculated as the difference of the time stamps. At that time Internet, in its early stage, was not so popular. They chose campus network instead of the worldwide network. They measured three connections in the campus network of different length. The measurements showed that 1) Delay was bigger at busy time of the day; 2) Routers could introduce significant delays.

Our measurement device also uses UDP and uses similar methods in measuring the network delay. VoIP device used in our measurements are explained in the next chapter.

2.3 RIPE NCC Test Traffic Measurements Service

IP address space is distributed in a hierarchical way. The Internet Assigned Numbers Authority (IANA) allocates blocks of IP address space to Regional Internet Registries (RIRs). RIRs allocate block of IP address space to Local Internet Registries (LIRs) that assign the addresses to End Users. The RIPE NCC (Réseaux IP Européens Network Coordination Center) is one of four RIRs that exist in the world today, providing allocation and registration services that support the operation of the Internet globally. The RIPE NCC performs activities primarily for the benefit of the membership in Europe, the Middle East, Central Asia and African countries located north of the equator. The services provided by the RIPE NCC ensure the fair distribution of global Internet resources in the RIPE NCC service region required for the stable and reliable operation of the Internet. This includes

the allocation of Internet (IP) address space, autonomous system numbers and the management of reverse domain name space.

RIPE (Réseaux IP Européens) is a collaborative forum existing since November 1989 and open to all parties interested in wide area IP networks. Work is carried out by individual volunteers in their own or their organisation's time. It was realized that not all activities could be carried out by volunteers, especially those needing continuity and constant availability. Therefore there was a need for the RIPE NCC to perform these coordinating tasks. A document describing the RIPE NCC was first published in September 1990, and the first activity plan was published in May 1991. RIPE asked RARE (Réseaux Associatés pour la Recherche Européenne) if they would provide the legal framework for the RIPE NCC. After a solicitation procedure, the RIPE NCC began in April 1992 with its headquarters in Amsterdam. Initial funding was provided by the academic networks (RARE members), European Academic and Research Network (EARN) and EUnet. Since it began the RIPE NCC has performed both administrative and innovative activities. The main administrative task is acting as a Regional Internet Registry which allocates and assigns IP address space and other Internet numbers. The innovative activities span a wide range from pioneering MBONE and WWW services to playing a major part in the development of the Internet Routing Registry (IRR). The RIPE NCC currently supports approximately 3200 LIRs that collectively form the RIPE NCC membership. Membership is open to anyone using the RIPE NCC services, primarily made up of Internet Service Providers (ISPs), telecommunication organizations and large corporations [15].

The RIPE NCC also provides services for the benefit of the Internet community at large including the development and maintenance of the RIPE database, administrative support for the RIPE community, and the development and co-ordination of new projects. In 1997, the RIPE NCC started Test Traffic project, a research to investigate the possibility of doing one-way delay and one-way loss measurements between sites. When it became clear that these measurements can be

reliably done on a large scale, the project was turned into Test Traffic Measurement (TTM) service. Since then, RIPE NCC is active in the field of active measurements with its TTM service [15].

In order to take one-way delay and one-way loss measurements RIPE NCC designed a test-box. The test box is an active measurement device that generates its own traffic and does not rely on packets from other sources. Traffic generated by the test-box for the measurements resembles the real traffic as much as possible, in order to make it hard to give the measurement traffic a preferential treatment. In order to satisfy this, ICMP-based techniques, such as the ping program, are not used in the test box, because many routers treat these packets differently from UDP or TCP traffic.

The general set-up of TTM is shown in Figure 2.1. To measure one-way delays between ISP-A and ISP-B, test-boxes are installed near the border routers of these ISPs. To measure one-way delays from ISP-A to ISP-B, the test-box at ISP-A starts to send packets to the test-box at ISP-B. Test-box in ISP-A adds the sending timestamp of each packet, and test-box in ISP-B saves the receive timestamp of each receiving packet. Then test-box in ISP-B can easily calculate one-way delay of each packet provided that the clocks on both sides are synchronized to the same clock. In order to accomplish this, the test-boxes are equipped with a GPS receiver. The TTM test-box is shown in Figure 2.2. The accuracy of the one-way delay measurement is about 10μs [16].

Beside delays the boxes also measure packet losses by counting the number of packets that were sent and received for the delay measurements. The route of the packets is determined by running the “traceroute” program at regular intervals. The entire setup is controlled by a central machine at the RIPE NCC, shown in the bottom of Figure 2.1. This machine also takes care of data-analysis, presentation of the results to the users, software maintenance and other tasks [16].

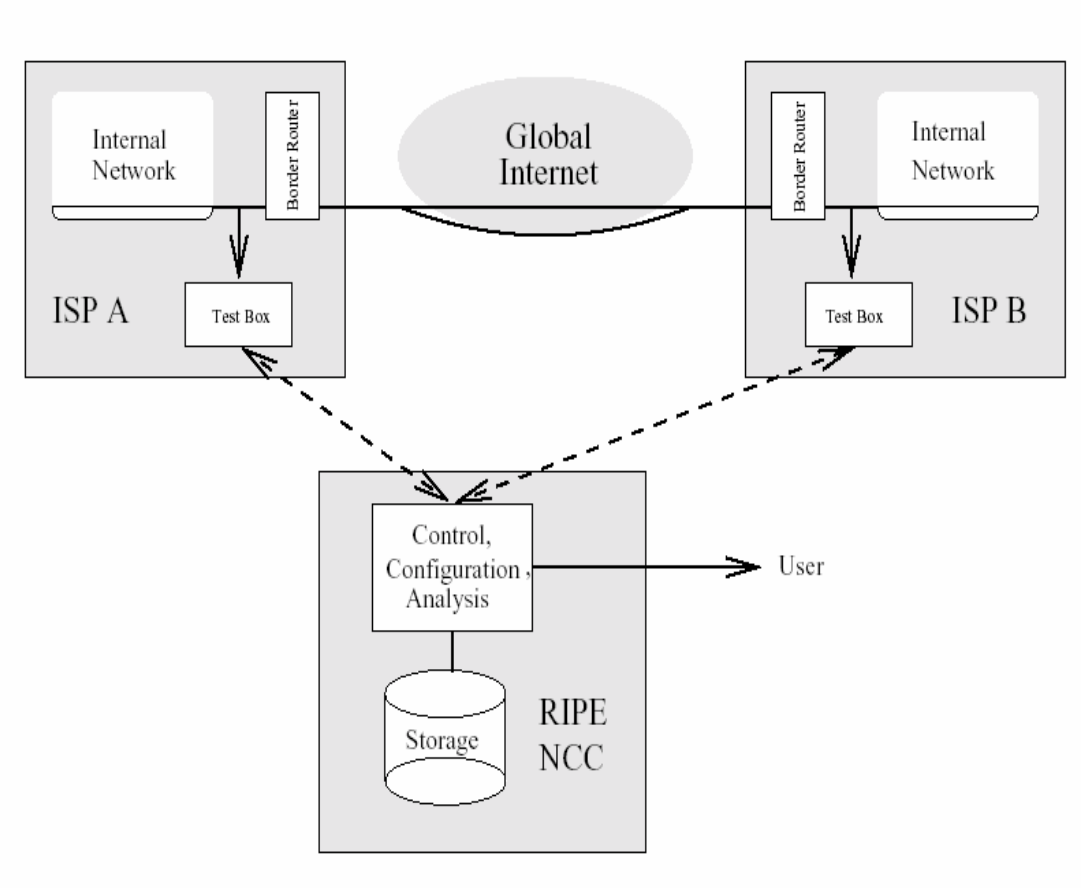

Figure 2.1 RIPE NCC TTM measurement setup

Figure 2.2 RIPE NCC TTM test-box

At the moment, the network of test-boxes consists of approximately 60 machines at locations all over the world, see Figure 2.3. On a typical day, approximately 40 to 50 boxes are operational. Each box sends traffic to each other box, thus providing information about 1600 to 2500 possible paths on the Internet [15].

Figure 2.3 Map Showing the Location of the Test-boxes Currently Installed. The Boxes Installed in New Zealand and Australia are not Shown.

Results of an analysis of end-to-end delay measurements performed by RIPE NCC TTM service are presented in [17]. In RIPE measurement configuration, fixed size IP probe packets of 100 bytes are sent from a source to a destination test-box. Packets are sent at a period of 40 seconds corresponding to 2160 packets per day. In total 963 normalized delay distributions have been taken into account. It is observed that, one-way delay distributions can be classified into 5 classes.

Class A : Gamma-like shape with heavy tail. (84 %)

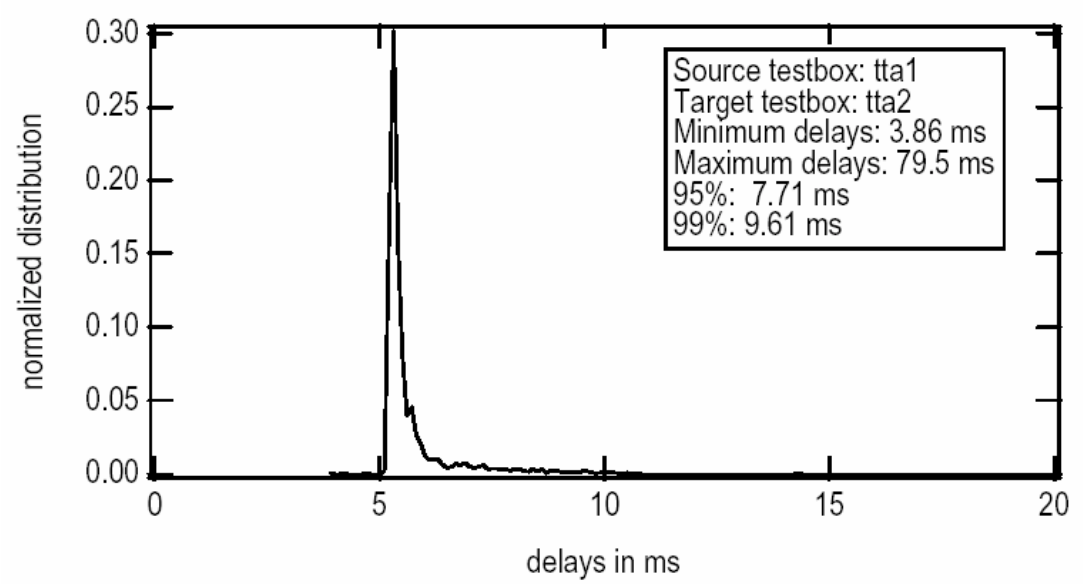

Figure 2.4 An example of class A distributions

Class B : Gamma-like with Gaussian or triangular lob (6.3 %)

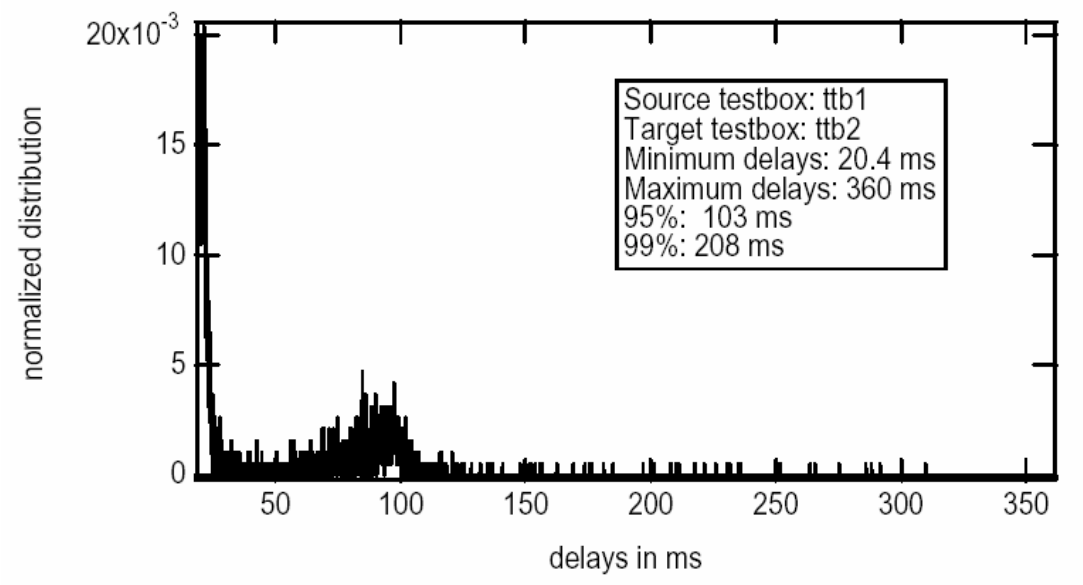

Figure 2.5 An example of class B distributions

Class C : 2 Gamma-like distributions (2.8 %)

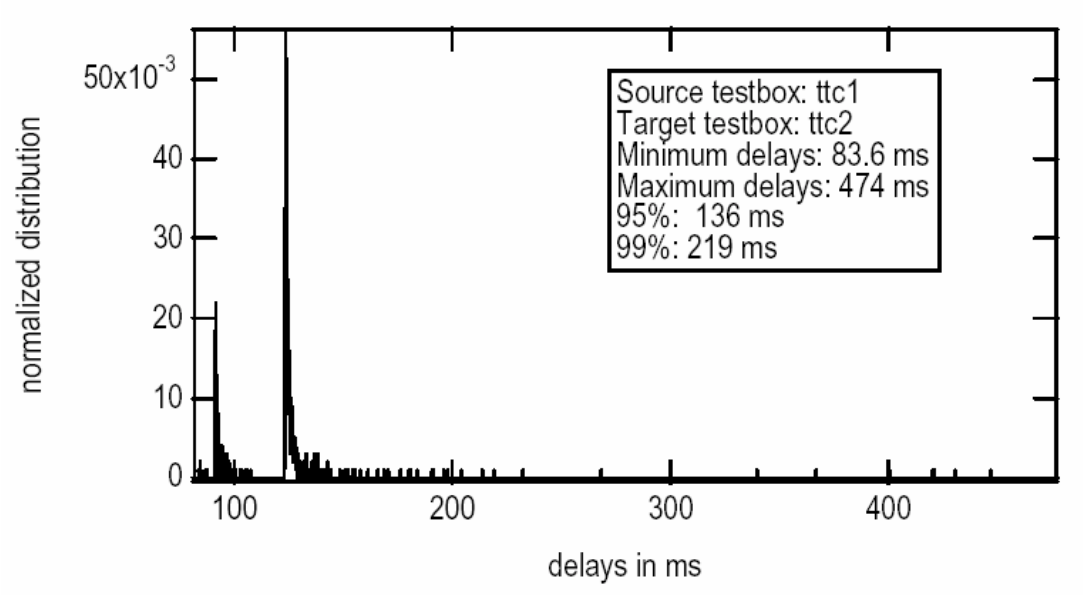

Figure 2.6 An example of class C distributions

Class D : Many peaks (5 %)

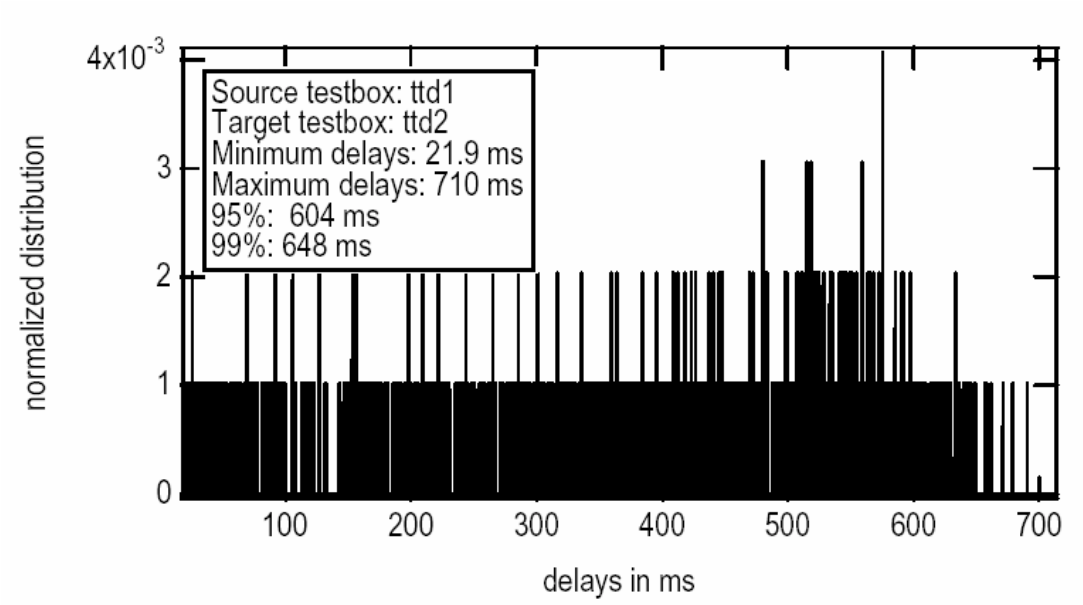

Figure 2.7 An example of class D distributions

Rest Class : Uncategorized (1.5 %). Most of them only occurred once and do not fit any of the classes described above.

We have achieved 37 one-way delay measurements and 42 RTT measurements on the path GIT-METU. For one-way delay measurements, fixed size IP probe packets of 135 bytes are send with a period of 16 ms. For RTT measurements, fixed size IP probe packets of 128 bytes are send with a period of 50 ms. The difference at packet size in one-way network delay and RTT measurements is because of a bug at the software of the device. In one-way delay measurements 128 bytes of voice data with 7 bytes header, corresponding to 135 bytes data are sent. During RTT measurements voice packets are not generated and we can adjust the packet length at RTT measurements via the user interface. If we select 128 bytes via the user interface, random content data packets are generated until total packet length including 7 byte header reaches to 128 bytes. We have observed that all of the distributions observed in one-way delay measurements fall into Class A distributions specified in [17]. For RTT measurements, 39 of 42 distributions fall into Class A, 1 distribution fall into Class B and 2 distributions have different shapes. The results of our measurements are mentioned in Chapter 5.

CHAPTER 3

MEASUREMENT TOOL ARCHITECTURE

3.1 Introduction

In this thesis, network delay and RTT measurements are achieved by a VoIP device, which has network delay measuring ability besides voice communication. VoIP device shown in Figure 3.1 is developed parallel to this thesis, by T. Çelikadam [18]. Device performs connection establishment, voice packetization, voice activity detection (VAD) and adaptive play-out buffering to combat network delay jitter. It establishes communication over a network using User Datagram Protocol (UDP). Device also has the ability to measure network delay observed by the received packets, number of lost packets and RTT. In this chapter this device is briefly explained. Detailed information about the device can be found in [18].

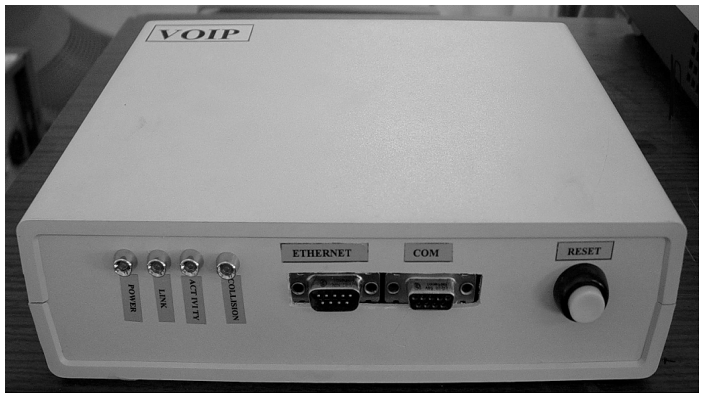

Figure 3.1 VoIP device used in measurements

3.2 Hardware

VoIP board is shown in Figure 3.2.

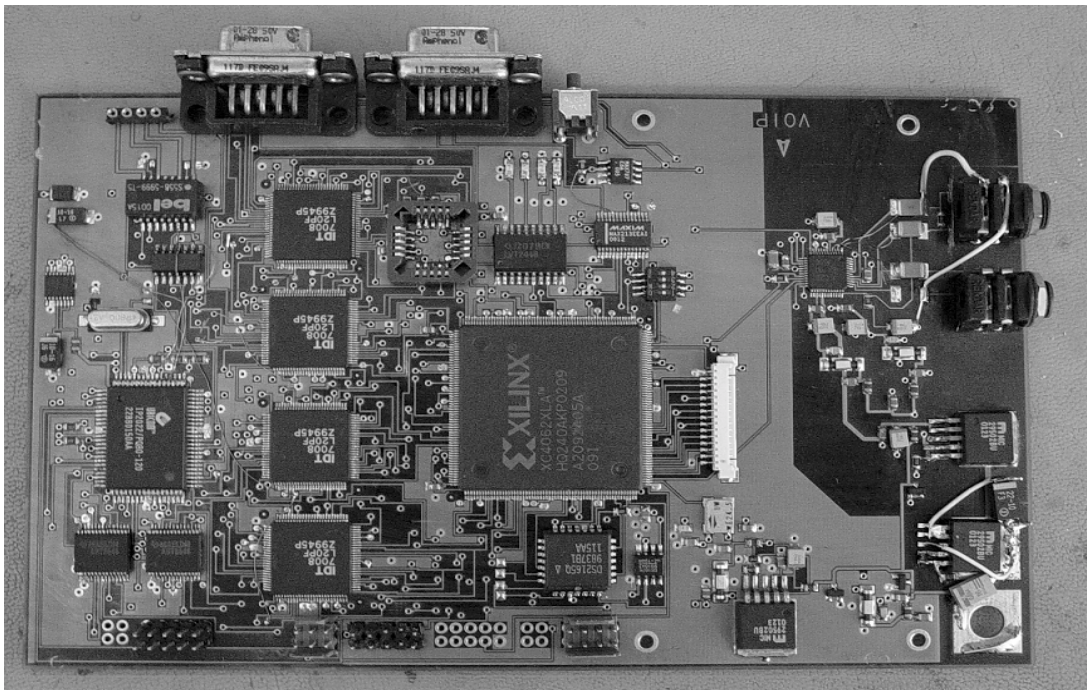

Figure 3.2 VoIP board

Analog-to-Digital conversion of analog voice signals is required in order to send the voice through the Internet. Also Digital-to-Analog conversion is required for received voice packets. These conversions are achieved by Texas Instruments TLV320AIC22 voice CODEC at the device. CODEC is controlled by Xilinx Inc. XC4028XLA Field Programmable Gate Array (FPGA) via CODEC's serial interface. FPGA gets the Analog-to-Digital conversion results from the CODEC via CODEC's serial interface. Also, FPGA sends the packets received from the Internet to the CODEC for Digital-to-Analog conversion via CODEC's serial interface. FPGA design is performed using the Very Large Scale Integrated Circuit Hardware Description Language (VHDL).

Four 16Kx32 dual-ported RAM (DPRAM) are used for play-out buffer memory requirements and to exchange voice samples between FPGA and microprocessor asynchronously. These four DPRAMs are connected as a single 64Kx32 DPRAM. FPGA and microprocessor communicates via DPRAMs and interrupt signals. DPRAM has two ports, which are completely independent of each other. Each of the ports has address bus width of 16 bits and data bus width of 32 bits. All memory locations are available to the both of the ports. Microprocessor uses one of the ports to read/write data to the DPRAM and FPGA uses the other port. Therefore, there is no direct connection between microprocessor and FPGA, so, data exchange is performed by memory read/write operations. ADC data is written by the FPGA to the DPRAM ADC data section and it is read by the microprocessor. DAC data, received from the network is written to the DAC data section and it is read by the FPGA to play the received audio. Two interrupt lines exist between FPGA and microprocessor to inform each other that the data is ready at the DPRAM.

Microprocessor is the Ubicom's IP2022 processor, which is also called Internet processor. Its high operating speed (120 MHz) minimizes the delay introduced on voice over IP communication by the application software. Microprocessor has a 10 Mbit Ethernet interface to send and receive voice packets. Ethernet interface is compliant to the IEEE 802.3 standard. Microprocessor has a serial RS232 UART port for transmitting diagnostics messages and receiving commands and device settings. Microprocessor takes the voice samples from the FPGA via DPRAMs and sends them to the receiving host using Ethernet interface. It can perform Voice Activity Detection (VAD) on voice samples if VAD option is enabled by the user. A play-out buffering algorithm, which adjusts the play-out times of the incoming voice packets according to network characteristics, runs on the microprocessor.

User interface of the device is provided by the serial interface of the microprocessor. Microprocessor communicates with the user interface program running on a Windows based PC, which is connected to the device via a serial port. User Interface of the device is explained in section 3.4.

Block diagram of the VoIP board is given in Figure 3.3.

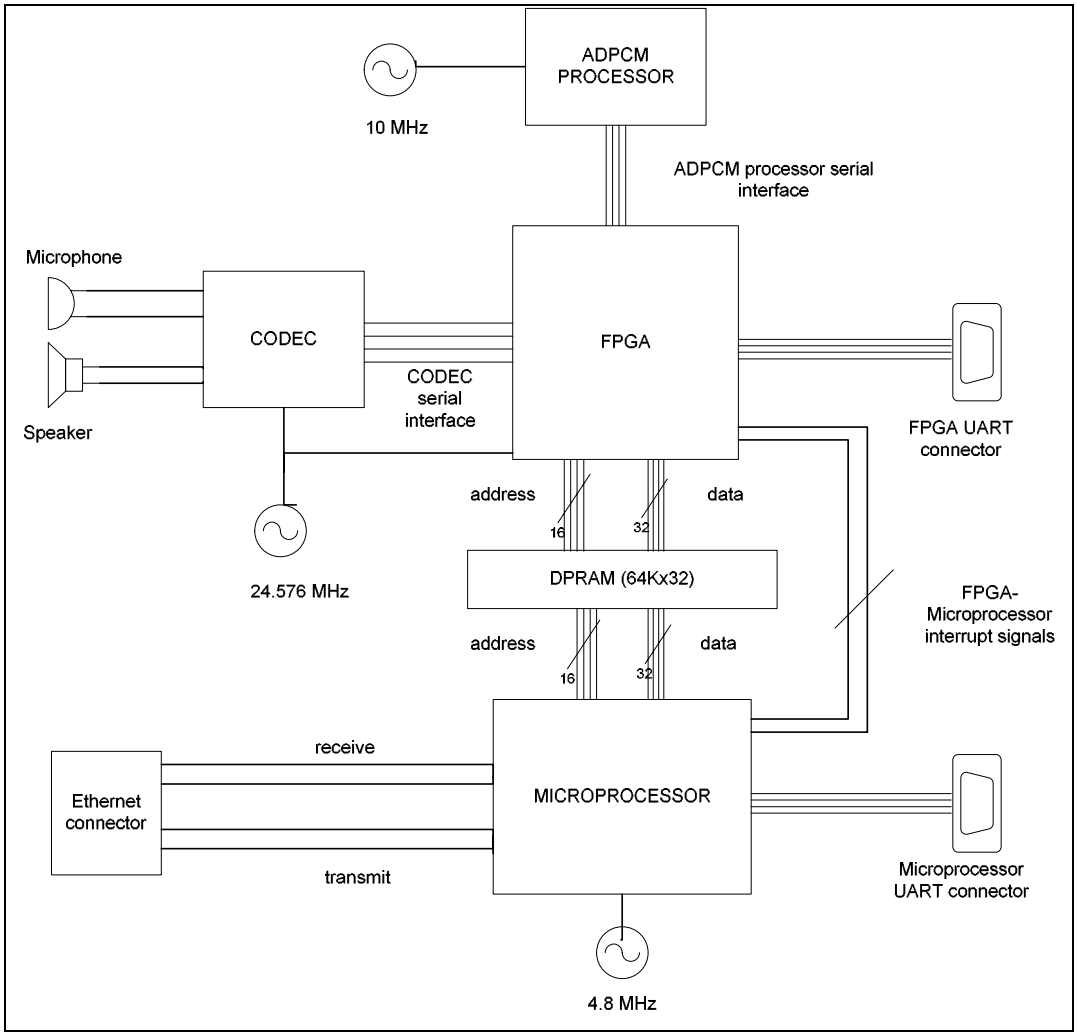

Figure 3.3 Block diagram of the VoIP board

3.3 Protocols

IP is a connectionless, packet-switched technology for carrying data across the Internet. It forms the network layer of the Internet and its primary function is to route packets from their source to their destination. For each packet, IP (Version 4, IPv4) adds a header with the format shown in Figure 3.4

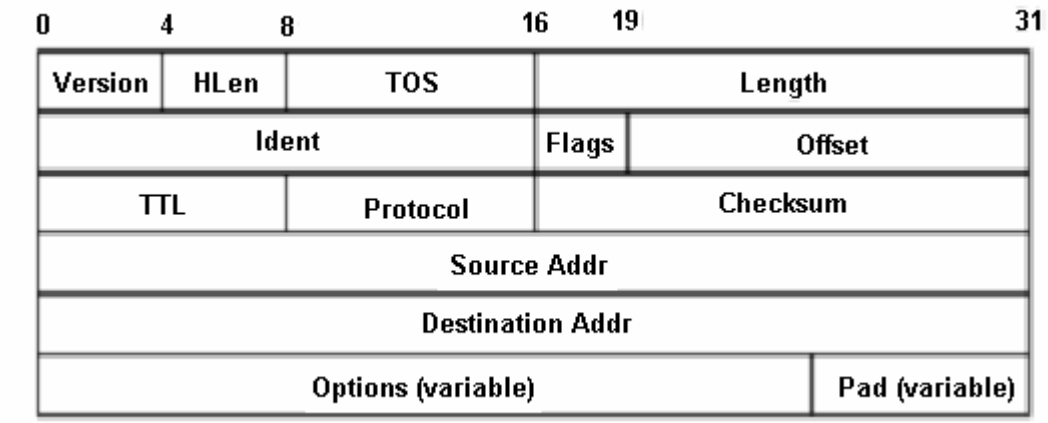

Figure 3.4 IP header

The Version field specifies the version number of IP. The HLen field specifies the length of the header in 32 bit words. Usually, the header length is 20 bytes. In cases where the Options field is used, it may be greater. The Type of Service (TOS) field allows packets to be classified and treated according to the class to which they belong. The Length field specifies the size of the payload in bytes. The Time to Live (TTL) field is actually a counter that counts the number of hops traversed by the packet. The Protocol field identifies the higher layer protocol whose packet is encapsulated within this packet. The Checksum field provides a checksum over the header. Fields containing the source and destination addresses of the packet are also included in the header. The Options field is used to provide some additional options and the Pad field is used to make the header size equal to an integral number of 32-bit words.

User Datagram Protocol (UDP) is a transport layer protocol that runs on top of IP. UDP provides connectionless host-to-host communication path. UDP has minimal overhead; each packet on the network is composed of a small header and user data, which is called a UDP datagram. Connectionless means that a datagram can be sent at any moment without prior advertising, negotiation or preparation. To transmit data, the UDP module simply passes the datagram to the internet layer which then

sends the datagram on its way. This means that just like IP itself, UDP is a best effort service. No guarantees about delivery are given, datagrams can get reordered and datagrams can be duplicated. Since the service which UDP offers is almost identical to the service of IP itself, it is possible for applications to send UDP datagrams to a multicast address and to receive UDP datagrams from a multicast group. The main advantages for UDP are that you can broadcast and it is fast. The main disadvantage is unreliability and therefore complicated to program at the application level. UDP uses a checksum to ensure the correctness of the message, but does not implement flow control, congestion control, reliable data transfer or in-order delivery of packets. It is more suited for real-time applications that are tolerant to losses but have stringent delay requirements. The UDP header is shown in Figure 3.5. The header contains the source and destination ports, which identify the sending and receiving applications. Next, it contains the number of data bytes which must be sent and finally the header contains space for an optional checksum. The exact specification of UDP can be found in [19].

Source Port	Destination Port
UDP Length	UDP Checksum

Figure 3.5 UDP header

Transmission Control Protocol (TCP) is a versatile transport layer protocol that finds widespread use in the Internet. TCP is a connection-oriented protocol that is responsible for reliable communication between two end processes. Being connection-oriented means that before actually transmitting data, you must open the connection between the two end points. The data can be transferred in full-duplex (send and receive on a single connection). When the transfer is done, connection must be closed. Both ends know when the session is opened (begin) and closed (end). The data transfer can not take place before both ends have agreed upon the connection. The connection can be closed by either side, but the other is notified. TCP enables multiplexing and ensures correctness the same way as UDP does. In

addition, it also provides flow control, congestion control, and reliable, in-order delivery of packets from sender to receiver. Unlike UDP, TCP guarantees delivery of data and also guarantees that packets will be delivered in the same order in which they were sent. It is suited for non real-time applications that require guaranteed delivery of data. It is not suitable for real-time applications. For a complete specification of TCP, you should refer to [20].

Table 3.1 lists popular Internet applications and the transport protocols that they use. E-mail, remote terminal access, the Web and file transfer run over TCP because these applications need the reliable data transfer service of TCP. Nevertheless, many important applications run over UDP rather TCP. UDP is used for RIP routing table updates, because the updates are sent periodically, so that lost updates are replaced by more up-to-date updates. UDP is used to carry network management (SNMP) data. UDP is preferred to TCP in this case, since network management must often run when the network is in a stressed state - precisely when reliable, congestion-controlled data transfer is difficult to achieve. Also, DNS runs over UDP, thereby avoiding TCP's connection establishment delays.

Table 3.1 Popular Internet applications and their underlying transport protocols.

Application	**Application-layer Protocol**	**Underlying Transport Protocol**
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	typically UDP
Streaming multimedia	proprietary	typically UDP
Internet telephony	proprietary	typically UDP
Network Management	SNMP	typically UDP
Routing Protocol	RIP	typically UDP
Name Translation	DNS	typically UDP

The real-time transport protocol (RTP) is an application layer protocol for multimedia traffic. It provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Those services include payload type identification, sequence numbering, timestamping and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services; both protocols contribute parts of the transport protocol functionality. However, RTP may be used with other suitable underlying network or transport protocols. RTP supports data transfer to multiple destinations using multicast distribution if provided by the underlying network. RTP itself does not provide any mechanism to ensure timely delivery or provide other quality-of-service guarantees, but relies on lower-layer services to do so. It does not guarantee delivery or prevent out-of-order delivery, nor does it assume that the underlying network is reliable and delivers packets in sequence. The sequence numbers included in RTP allow the receiver to reconstruct the sender's packet sequence, but sequence numbers might also be used to determine the proper location of a packet, for example in video decoding, without necessarily decoding packets in sequence. RTP protocol is adequate for Internet Radio, Internet Telephony (VoIP), video-on-demand, and other multimedia applications. For a complete specification of RTP, you should refer to [21].

Transport protocol of the VoIP device is User Datagram Protocol (UDP). VoIP device has a modified RTP transport protocol, explained in [18]. This protocol, as RTP, seats on top of UDP header. Unused parts of RTP header are removed and this modified RTP protocol is obtained. Designed VoIP devices can only communicate with each other, because of the modified RTP protocol. Modified RTP header is given in Figure 3.6.

Packet Type Identification Character (PTI) 1-Byte	Sender Timestamp (Ts) 4-Byte	Sequence Number (Sn) 2-Byte

Figure 3.6 RTP header

Sequence numbers are used for packet sequencing and loss detection in the receiver. Timestamps are used to determine the delay experienced by the packet while traversing the network. The timestamps are relative to the start of the stream. Timestamps allow user to determine play-out time at adaptive play-out buffering applications. Play-out buffering reduces the effects of delay jitter on the played out speech. An additional byte is used at the beginning of the RTP header for packet type identification and control purposes, which is referred as packet type identification character. Receiver of the packet starts some tasks according to the received packets packet type identification character [18].

Network interface of the VoIP device is the Ethernet, which is the most popular Local Area Network (LAN). After RTP packet is generated, it is embedded in UDP packet and UDP packet is embedded into the IP packet. If the device is on an Ethernet network, as in our case, IP packets are then put in Ethernet frames for transmission. This packet nesting is shown in Figure 3.7.

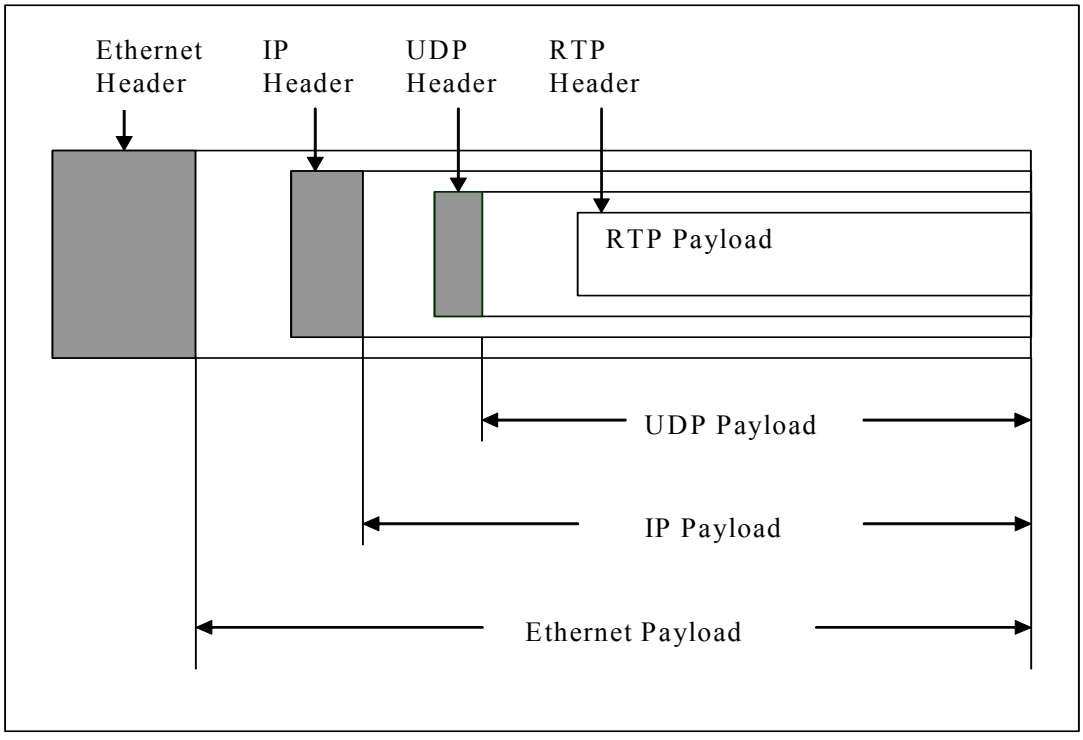

Figure 3.7. Packet nesting

3.4 User Interface Program

User interface program is used for controlling the VoIP device and extracting data from the VoIP device. COM1 port of the PC must be connected to the serial port of the VoIP device for proper operation. User Interface Menu is shown in Figure 3.8.

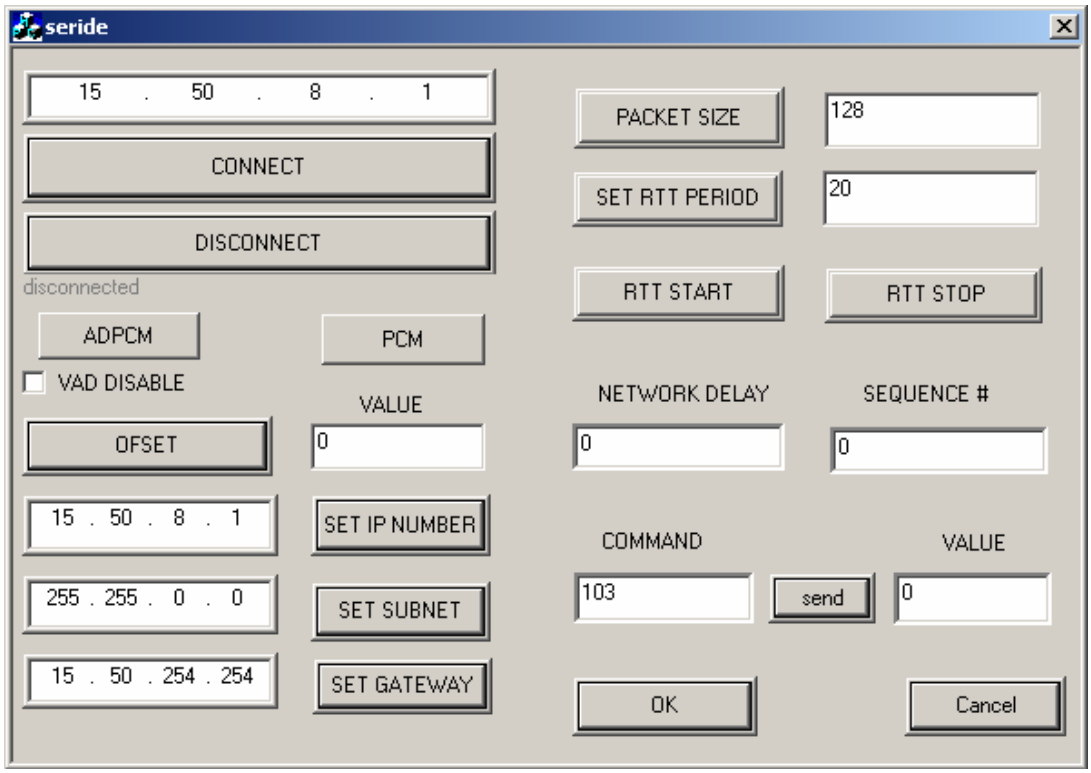

Figure 3.8 User interface program

User interface has the following features;

Connection establishment to a host (By setting the IP address of the host),

Adjusting device settings due to network (IP Address, Gateway, Subnet Mask)

Enable/Disable Voice Active Detection (VAD)

Calculate clock offset command

Start/Stop RTT measurement

Adjusting packet size and period for RTT measurements

Display Network Delays / RTT values and packet numbers

When device is inserted in a network, its IP number, subnet mask and gateway numbers must be assigned according to the inserted subnetwork. These settings can be done using the buttons named *SET IP_NUMBER, SET SUBNET, SET GATEWAY* and their related edit boxes located at the left of the corresponding buttons as shown in Figure 3.8. The important point while setting these numbers is that, button must be clicked after the number is written to the corresponding edit box. After network number settings are done, device is ready to use for VoIP communication.

Connection can be established with the host using the user interface program. Before connecting to a host clock offset with the host clock should be removed. For this purpose host IP number should be written to the edit box at the top left corner of the window. After host IP number is written, *CLOCK OFFSET* button should be clicked. Measured clock offset value is returned to the box, at the right side of the clock offset button. This value is also recorded to the log file. Detailed information about clock offset is given in chapter 4.

After clock offset is removed, connection can be established with the host by clicking the CONNECT button. When connection is established, diagnostic message showing the status of the connection changes from *disconnected* to *connected.* This message is located below the disconnect button.

When connection is established, VoIP device returns the sequence number and measured one-way network delay of each received packet to the user interface program. User Interface program simultaneously displays this values at the boxes named *Sequence #* and *Network Delay*. These values are also written to the log file.

User can end the connection by clicking the disconnect button. When connection is released, diagnostic message showing the status of the connection changes from connected to the disconnected. It is important to state that, the delay values measured during connection is not the end-to-end delay but the network delay.

Voice Activity Detection (VAD) feature is disabled by default. By clicking the VAD Enable check box, VAD can be enabled. During conversations people do not talk continuously, silence periods occur when sender stops talking. Sending packets during this period means loading the network for unused information. When VAD is enabled, VoIP device senses whether or not the user is speaking and sends voice packets only when user is speaking as mentioned, and does not send any packets during silence periods. User can disable the VAD by clicking the check box again.

User can start RTT measurement and set RTT measurement parameters. User can set size of the packets used in the RTT measurement. In order to do this, the packet size value must be written to the edit box first, which is at right of the *PACKET SIZE* button, and then the *PACKET SIZE* button must be clicked. Unit of the packet size is byte and default value of the packet size is 128 bytes. User can set the time difference between departures of the two successive packets used for RTT measurement. In order to do this, the time difference value must be written to the edit box, which is at the right of the *SET PERIOD* button, and then the *SET PERIOD* button must be clicked. Unit of the RTT packet generation period is in milliseconds and default value of the RTT packet generation period is 20 milliseconds. After RTT measurement parameters are set, user can start RTT measurement by clicking the *RTT Start* button and end measurement by clicking the *RTT Stop* button. Measurement results returned from VoIP device are displayed to the *Sequence #* and *Network Delay* boxes simultaneously and also written to the log file.

CHAPTER 4

MEASUREMENTS

4.1 Introduction

End-to-end delay and loss traces are frequently used in analyzing network performance. The accuracy of such measurements is important for several reasons. First, end-to-end measurements may be the only way of measuring network performance, especially when there is no provision inside the network to provide end-systems with information about the current status of the network. The current Internet has no mechanism for providing feedback on network congestion to end-systems at the IP layer. Second, protocols and applications that behave adaptively at the end-system base their control on observed network performance, and it is critical that they obtain correct measurements. Understanding the packet delay and loss behavior of the Internet is important for proper design of network algorithms such as routing and flow control algorithms, for the dimensioning of buffers and link capacity, and for choosing parameters in simulation and analytic studies. It is also essential for designing the emerging audio and video applications. For example, the shape of the delay distribution is crucial for the proper sizing of playback buffers [22].

Measurement setup is shown in Figure 4.1. Two VoIP devices are used in order to measure RTT, one-way delay and packet loss between two sites. VoIP device has Ethernet interface, so we must locate the VoIP devices to LANs for measurement. We have located one device at Electrical and Electronics Engineering Department

of METU. We have measured one-way delay and RTT on two paths. For the first path, measurements at 28-29 June 2003, we have located the other device at Electrical and Electronics Engineering Department of HU. We have controlled the measurement by the device located in HU, so, measured one-way delays were from METU to HU. For the second path, measurements at 27 October to 14 November, we have located the other device at GIT. We have controlled the measurement by the device located in METU, so, measured one-way delays were from GIT to METU. Before measurements, fixed IP addresses are provided for both VoIP devices and they are introduced to the LANs with these IP addresses. These IP addresses, subnet mask and gateway numbers are introduced to the VoIP devices via the user interface.

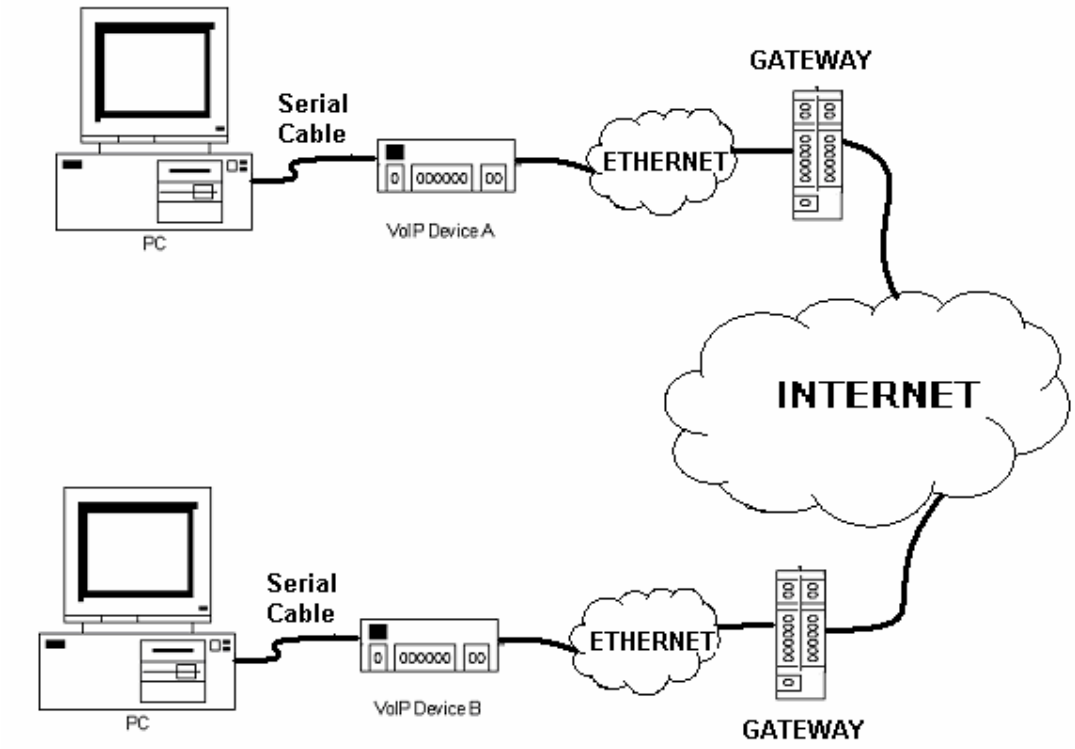

Figure 4.1 Measurement set-up

In this chapter measurement methods are explained, results of the measurements will be discussed in Chapter 5.

4.2 One-way Network Delay Measurements

One-way network delay is important in voice communication, particularly if it is not equal in each direction. One-way network delay measurement involves two clocks, so, the synchronization between these two clocks has a tremendous impact on the accuracy of the measurement. Measuring the one-way delay of network connections without the use of synchronized clocks is a non-trivial task. Many methods rely on RTT measurements and halve the result, hence estimating the one-way delay. We did not use this method and attempted to eliminate the effects of unsynchronized receiver and transmitter clocks in order to obtain correct delay values.

For delay measurements, a sender needs to add timestamps to packets for a receiver to gather delay information. Since the clocks at both end-systems are involved in measuring delay, the synchronization of the two clocks becomes an issue in the accuracy of delay measurement. The Network Time Protocol (NTP) [23] is widely used in the Internet for clock synchronization. It provides accuracy of the order of milliseconds under reasonable circumstances. The accuracy, however, is not guaranteed, and not all hosts on the Internet support it.

In this thesis, we have measured network delay by developed VoIP device. In order to measure one-way network delay, sender VoIP device assigns a timestamp for each packet leaving and attaches this timestamp at RTP header of the leaving packet, in order to provide the receiver VoIP device to calculate the delay. When the packet arrives at the receiver host, the delay is calculated using the receiver's clock. In this method, however, time clocks of the sender and receiver should be synchronized in order to measure accurate delays. When two clocks are not synchronized and, more specifically, have different frequencies, time duration measured with one clock will be different from the other. Unsynchronized clocks results offset and skew as illustrated in Figure 4.2.

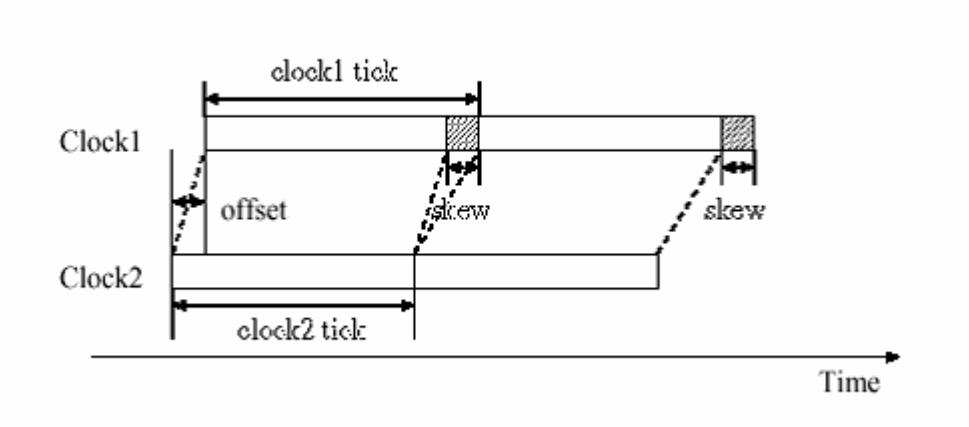

Figure 4.2 Clock offset and skew between transmitter and receiver clocks

The relative offset of the two clocks is caused when the two clocks have different time. The relative offset introduces a margin of errors in determining the one-way delay. Since, the relative skew is caused by the difference of tick intervals between two clocks, the one-way delay becomes linearly increased or decreased due to it. We say that a time duration measured with a clock is consistent with that by any other clock of the same frequency and any offset.

For one-way network delay measurements, the sender adds a timestamp to a packet when it leaves the sender and the receiver records the time the packet arrives at the receiver. When the two host clocks are perfectly synchronized, the difference between the two timestamps is the one-way network delay experienced by that packet. If the clocks on the two hosts have a non-zero offset, but no skew, the difference between two timestamps includes not only the one-way delay, but also the offset. Given only a one-way measurement, we cannot distinguish the offset from the measurement, unless we are given the network delay, which is what we intended to measure in the first place. If the clocks have a non-zero skew, not only is the one-way delay measurement off by an amount equal to the offset, but it also gradually increases or decreases over time depending on whether the sender clock runs slower or faster than the receiver clock.

Let C_A and C_B be two clocks, then;

Offset: The offset of the clock C_A relative to C_B at time $t \geq 0$ is $C_A(t) - C_B(t)$.

Frequency: The frequency of C_A at time t is $C_A'(t)$.

Skew: The skew of C_A relative to C_B at time t is $(C_A'(t) - C_B'(t))$.

Drift: The drift of C_A relative to $C_B(t)$ at time t ≥ 0 is $(C_A''(t) - C_B''(t))$.

Clock ratio: The ratio of C_A relative to C_B at time t is $C_A'(t) / C_B'(t)$.

Two clocks are said to be synchronized at a particular moment in time if both the offset and skew are zero. Let C_A and C_B have constant frequencies, and α and δ be the clock ratio and skew of C_B relative to C_A, respectively. $\alpha = C'_B / C'_A$ and $\delta = C'_B - C'_A$. Then the relation between the clock ratio and the skew is:

$$\delta = C'_B - C'_A = \alpha\, C'_A - C'_A = (\alpha - 1)\, C'_A \qquad (4.1)$$

We assume that the sender and receiver clocks have constant frequencies, so that their skew and clock ratio are constant over time.

4.2.1 Clock Offset Calculation

Paxson's approach is used in order to remove the effect of clock offset [10]. In Paxson's method, the network path between sender and receiver is assumed to be symmetric, so the network delay of both directions is assumed to be the same during offset calculation. This assumption leads us to calculate the clock offset easily.

Developed VoIP device measures one-way network delay during conversation. Before establishing the connection, offset calculation must be achieved by the user via user interface. For this purpose host IP number should be written to the edit box

at the top left corner of the window. After host IP number is written, *CLOCK OFFSET* button should be clicked. Measured clock offset value is returned to the box at the right side of the clock offset button and also recorded to the log file.

Let's call the requester VoIP device as Client A and the target VoIP device as Client B. As we have mentioned before, measurements are controlled via the User Interface of the requester VoIP device, Client A. When *CLOCK OFFSET* button is clicked, Client A constructs a UDP packet for clock offset calculation request and sends it to the Client B. First byte of the UDP packet, the packet type indicator (PTI) byte, indicates that this packet is a clock offset calculation request packet. Following four bytes includes the timer value of the microprocessor at the UDP packet's construction instant. Let's call this timer value as TA_s, send time of Client A.

When Client B receives clock offset calculation request packet, it immediately reads the value of its timer and adds this value to the end of the received UDP packet's data section. Let's call this timer value as TB_r, receive time of Client B. Then, Client B changes the packet type indicator byte in order to indicate that this packet is a clock offset calculation reply packet. Then, it reads its timer value again, that shows the send instant. Let's call this timer value as TB_s, send time of Client B. Finally, Client B writes this timer value to the end of the constructed packet and sends this reply packet to Client A.

When Client A receives clock offset calculation reply packet, it reads its timer value immediately and stores it as receive time. Let's call this timer value as TA_r, receive time of Client A. Now, Client A has four timer values in order to calculate the clock offset.

UDP packet exchange for clock offset calculation between clients A and B is seen in Figure 4.3. Client A request clock offset calculation and client B replies it.

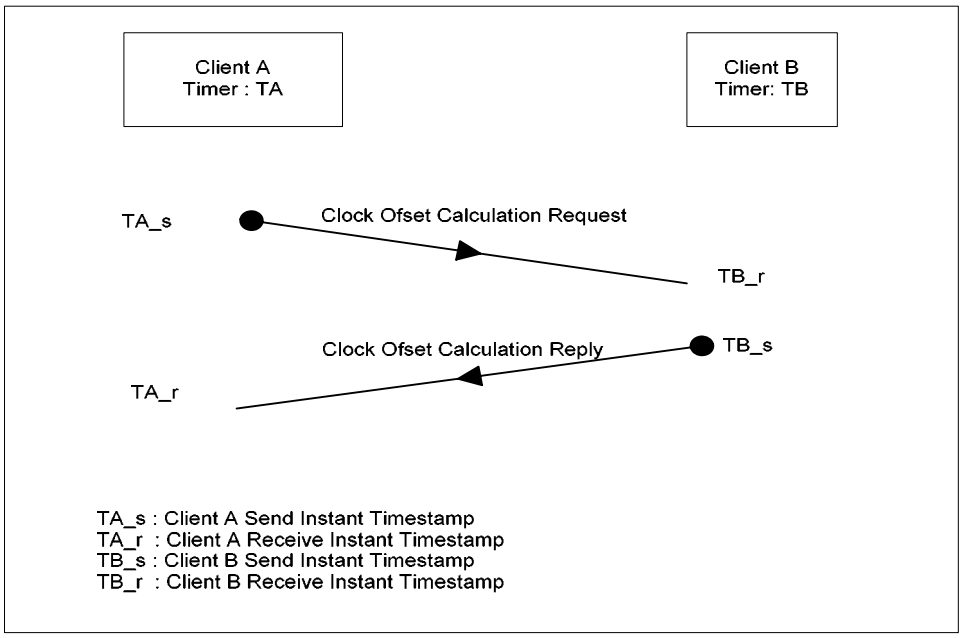

Figure 4.3: UDP packet exchange for clock offset calculation between clients A and B

It is assumed that, network delay observed by the request and reply packets are same. This assumption makes the calculation of clock offset possible. Let's says both of the packets observe network delay of *nd* milliseconds and timer value of the client B is higher than the timer value of the client A by ΔC ms offset. Then following equations can be written for receive times.

$$TB\_r = TA\_s + \Delta\mathrm{C} + \mathrm{nd} \tag{4.2}$$

$$TA\_r = TB\_s - \Delta\mathrm{C} + \mathrm{nd} \tag{4.3}$$

By manipulating above two equations, equation (4.4) is obtained for clock offset calculation.

$$\Delta C = \frac{(TB\_s + TB\_r) - (TA\_s + TA\_r)}{2} \tag{4.4}$$

Client A calculates the clock offset using equation (4.4) after the clock offset calculation reply packet is received. Measured clock offset value is returned to the information box, at the right side of the clock offset button. Client A uses this value at one-way delay measurements in order to remove the effect of clock offset between two Clients.

After clock offset value is calculated, connection must be established between Client A and Client B for one-way delay measurements. Connection can be established by clicking the CONNECT button at the user interface program of the Client A. When connection is established, diagnostic message showing the status of the connection changes from *disconnected* to *connected.* This message is located below the disconnect button.

When connection is established, Client A displays the sequence number and measured one-way network delay of each received packet at the boxes named *sequence #* and *Network Delay* at its User Interface. These values are also written to the log file. It is important to state that, the delay values measured during connection is not the end-to-end delay but the network delay.

After clock offset is calculated and connection is established, Client A and Client B starts to send voice packets to each other. A voice UDP packet data section is total of 135 bytes in length and consists of packet type identification byte, 2-byte sequence number of voice UDP packets, 4 byte timestamp information that is used to show the packets generation instant and 128 byte 8-bit μ-law compressed voice samples. Voice sampling frequency is 8 kHz, so, 128 voice samples carry voice information of 16 ms duration. This means that voice packets are generated with a period of 16 ms. Timestamps are used for delay calculation and the sequence numbers are used to arrange the received packets in the correct generation order. The sequence numbers are also used to detect lost packets.

As we have mentioned above, Client B adds packet send time (TB_s) to every packet it sends to client A. Client A records packet receive time of each received packet (TA_r). Now, Client A has the knowledge of receive time (due to its own clock) and send time (due to Client B's clock) of a packet, and clock offset value. Client A calculates one-way network delay by using equation (4.5) as;

$$\text{nd} = TA\_r - TB\_s + \Delta\text{C} \tag{4.5}$$

This one-way network delay is the delay from Client B to Client A. We have measured one-way network delay on two paths. For the first path, measurements at 28-29 June 2003, we have measured one-way network delays from METU to HU. For the second path, measurements at 27 October to 14 November, we have measured one-way network delays from GIT to METU. One-way network delay, measured on path2 at 10:10 AM at 30.10.2003, is plotted at Figure 4.4.

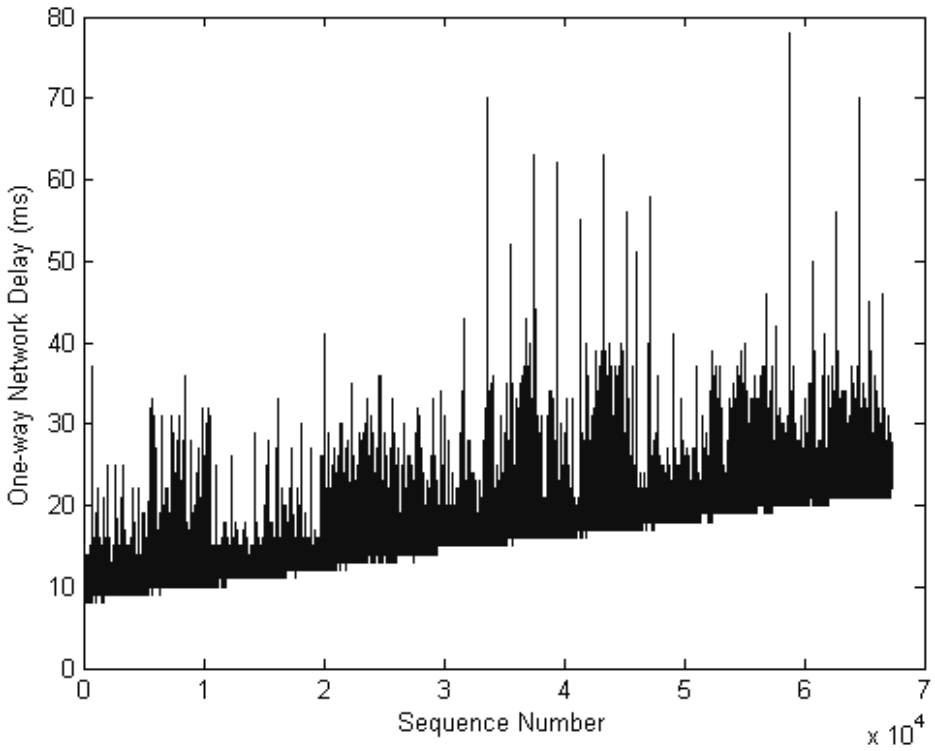

Figure 4.4 Measured one-way network delay

4.2.2 Clock Skew Elimination

If the clock ratio between the sender and receiver clocks is greater than or less than one, network delays will appear to become longer or shorter over the course of a measurement period as seen in Figure 4.4. In Figure 4.4, the one-way network delay shows an increasing trend. While one might imagine that this is due to increasing congestion and queuing delay, it is unlikely as the minimum observed delay increases over time. Instead, the linear increase in delay attests to a constant speed difference between the sender and receiver clocks, so to skew. This increasing trend distorts performance metrics such as the average and autocorrelation of one-way delay. The purpose of removing the effect of skew on the delay measurements is to transform the delay measurements so that they are consistent with the receiver clock.

For different size packets, the clock skew may not be distinguishable from the delay trend, if any. For example, if the packet size grows over time and the route from the sender to the receiver is fixed, then the transmission delay gradually increases, and it is hard to distinguish a skew from this delay trend. Thus we have used same size packets during measurements. Let us now introduce the terminology for clocks, timestamps, and delays used in measurements.

Cs : Sender Clock.

Cr : Receiver Clock.

N : Number of packets that arrive at the receiver.

l_i : timestamp of the i-th packet leaving the sender according to Cr, i = 1,2,…..,N

t_i^s : timestamp of the i-th packet leaving the sender according to Cs, i = 1,2,…,N; $t_i^s = Cs\ (l_i)$.

t_i^r : timestamp of the i-th packet arriving at the receiver according to Cs, i = 1,2,..,N

d_i : end-to-end delay measurement of the i-th packet, using timestamps t_i^s and t_i^r, i = 1,2,…..,N;

$$d_i = t_i^r - t_i^s \tag{4.6}$$

Figure 4.5 shows the timing difference between Cs and Cr when Cs runs at half the speed of Cr and all the packets experience the same network delay. The end-to-end delay of the i-th packet consistent with Cr is $t_i^r - l_i$. However, l_i is not known at the receiver and d_i is estimated using t_i^s and t_i^r . As a result, in this case, the end-to-end delay is consistent with neither Cs nor Cr. To make it consistent with Cr, we need to determine the skew of Cr relative to Cs, and remove it from the measurement d_i.

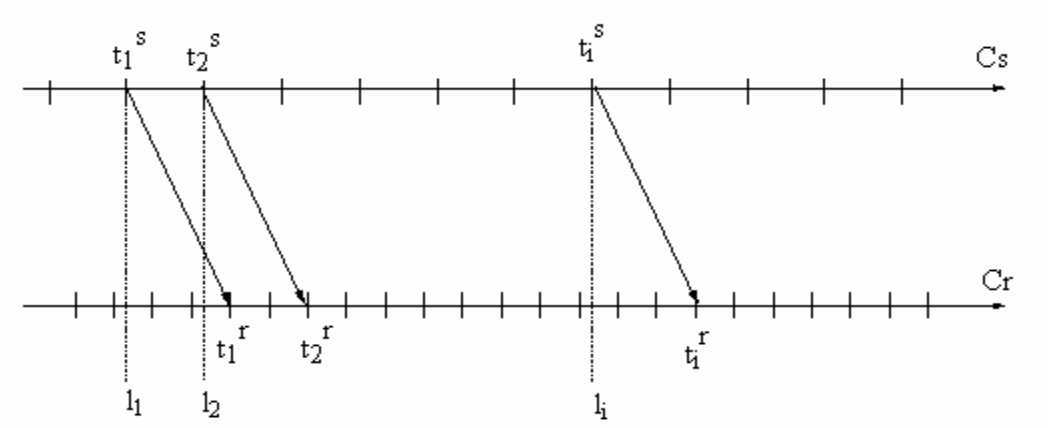

Figure 4.5 Timing chart showing constant delay

When there is a constant clock skew between two clocks, the clock offset between them gradually increases or decreases over time, depending on the sign of the skew. The amount of increase or decrease in the clock offset is proportional to the time duration of observation. We use the change in offset to estimate the clock skew. Thus it is more convenient to use timestamps relative to a specific point in time,

such as the departure or arrival time of the first packet, than to use absolute timestamps. Below relative timestamps at the sender and the receiver are introduced.

T_i^s : time duration between the first and the i-th packets' departures at the sender consistent with Cs, $\Delta(l_1 , l_i , Cs)$.

$$T_1^s = 0 \text{ and } T_i^s = \Delta(l_1 , l_i , Cs) \equiv t_i^s - t_1^s. \tag{4.7}$$

T_i^r : time duration between the first and the i-th packets' arrivals at the receiver consistent with Cr.

$$T_1^r = 0 \text{ and } T_i^r = t_i^r - t_1^r. \tag{4.8}$$

Then, using (4.1) we obtain,

$$\Delta(l_1 , l_i , Cr) = l_i - l_1 = \alpha\Delta(l_1 , l_i , Cs) = \alpha T_i^s . \tag{4.9}$$

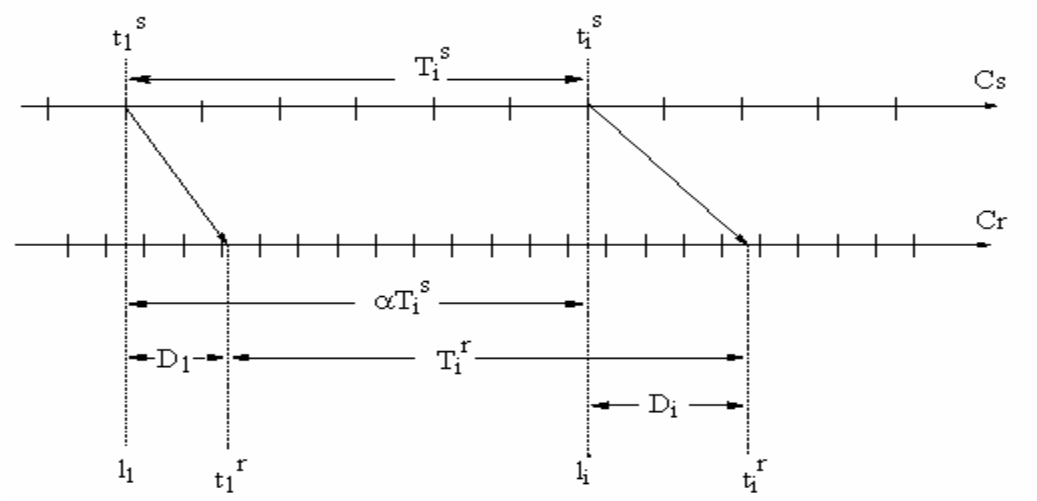

Figure 4.6 Timing chart showing variable delay

Relationship between $\Delta(l_1 , l_i , Cr)$ and T_i^s is illustrated in Figure 4.6. The quantities D_1 and D_i shown in the Figure 4.6 are defined as;

D_i : end-to-end delay consistent with Cr.

$$D_1 = \Delta(l_1 , t_1^r , Cr) = t_1^r - l_1 , \tag{4.10}$$

$$D_i = \Delta(l_1 , t_i^r , Cr) = t_i^r - l_i , \tag{4.11}$$

Using (4.9) in (4.11) we obtain,

$$D_i = t_i^r - l_1 - \alpha T_i^s \tag{4.12}$$

Adding and subtracting t_1^r to the right side of (4.12) we obtain,

$$D_i = (t_i^r - t_1^r) + (t_1^r - l_1) - \alpha T_i^s = T_i^r + D_1 - \alpha T_i^s \tag{4.13}$$

The quantity D_i, however, is not obtainable directly from actual timestamps, due to the skew between the sender and the receiver clocks. The quantity that is obtainable from actual timestamps is the following.

D_i : end-to-end delay calculated from T_i^s and T_i^r.

$$D_i = T_i^r - T_i^s \tag{4.14}$$

Adding and subtracting D_1 and αT_{is} to the right side of (4.14) we obtain,

$$D_i = T_{ir} + D_1 - \alpha T_{is} + (\alpha - 1) T_{is} - D_1 = D_i + (\alpha - 1) T_{is} - D_1 \tag{4.15}$$

The goal of estimating and removing the clock skew is to obtain D_i from actual delay measurements D_i. Note that D_i differs from D_i by $(\alpha - 1)\, T_i^s$ minus a constant D_1. If $\alpha > 1$ then, $(\alpha - 1)\, T_i^s$ grows linearly with T_i^s and D_i gets larger. This is the reason of the increasing trend we observed in Figure 4.4. We can obtain D_i from actual delay measurements D_i by the formula,

$$D_i = D_i - (\alpha - 1)\, T_i^s + D_1 \tag{4.16}$$

We must estimate the values of α and D_1 in order to obtain D_i from D_i by using (4.16).

4.2.2.1 Skew Estimation Algorithms

There are several algorithms introduced to estimate the skew in one-way network delay measurements. Consider the one-way network delay measurement illustrated in Figure 4.4. We can determine the skew by drawing a line, which passes below all the delay values in the y-axis but as close as possible, by a ruler. Then, we measure the angle between the line and the x-axis, and calculate the skew using simple trigonometry. This approach is hard to automate, and invites human errors. A second approach would be to determine the first and last data points and draw a line between them. But we can easily see that this approach does not give accurate results because of the variability of the network delay.

Moon's approach is used in order to eliminate skew from our measurements [5]. Moon's approach is to fit a line that lies under all the data points, but as closely to them as possible. He has formulated this idea as a linear programming problem. The two objectives of the algorithm are; first, the line should lie under all data points, second, the sum of distances between the line and all the data points on the y-axis

must be minimized. We have constructed our algorithm considering these two objectives. There are also other algorithms such as, Linear Regression Algorithm, Piecewise Minimum algorithm and Paxson's algorithm. Moon has compared his algorithm with these algorithms [5]. We have chosen Moon's algorithm considering this comparison and implemented the algorithm by MATLAB.

Figure 4.7 shows the measured one-way network delay on the path GIT-METU at 10:10 AM at 30 October 2003. The skew line estimated by the implemented algorithm is also shown. Mean value of network delay was 17.63 ms before skew removal and 11.4 ms after skew removal. Figure 4.8 shows the skew eliminated one-way network delay.

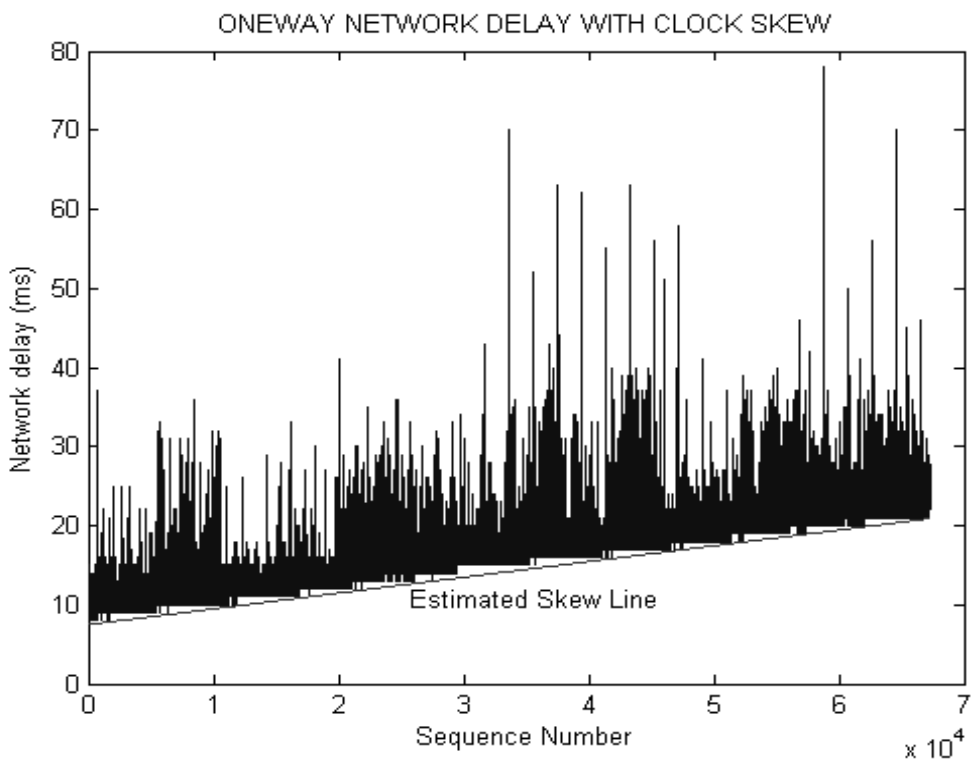

Figure 4.7 Measured one-way network delay and estimated skew line

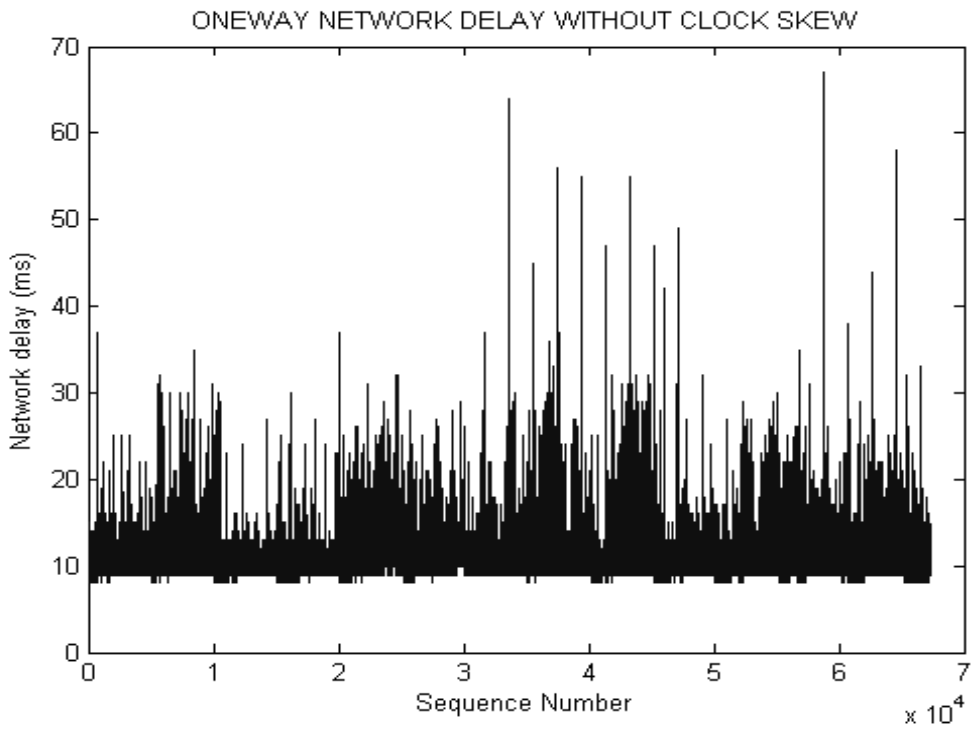

Figure 4.8 One-way network delay after skew removal

4.3 RTT Measurements

Round Trip Time (RTT) measurements are widely used in network analysis. In [12] and [24], small UDP packets are sent every 39.06 ms from a source node to a destination node, and echoed back to the source node. The authors show how their measurements can be used to detect problems in the Internet. For example, they observed in May 1992 that round trip time values would increase dramatically every 90 seconds. They identified the problem as being caused by a debug option in some gateway software.

We have used the same method as in [12] and [24], echoing UDP packets method, in our RTT measurements. RTT measurement ability is added to the developed VoIP device, used in our measurements. Unlike one-way network delay measurements, VoIP device does not send voice packets during RTT measurements.

It generates adjustable sized UDP packets at adjustable intervals during RTT measurements. By varying the UDP packet size and the interval between successive packets, we can examine the RTT behavior of the Internet over different time scales and loads.

Size of the UDP packets, send during RTT measurements, can be adjusted via the Serial User Interface of the requester VoIP device (Client A). In order to do this, the packet size value must be written to the edit box first, which is at right of the *PACKET SIZE* button, and then the *PACKET SIZE* button must be clicked. Unit of the packet size is byte and default value of the packet size is 128 bytes.

The interval between successive UDP packets, send during RTT measurements, can be adjusted via the Serial User Interface of the requester VoIP device (Client A). In order to do this, the interval value must be written to the edit box, which is at the right of the *SET PERIOD* button, and then the *SET PERIOD* button must be clicked. Unit of the RTT packet generation period is in milliseconds and default value of the RTT packet generation period is 20 milliseconds.

After RTT measurement parameters are set, user can start RTT measurement. For this purpose host IP number should be written to the edit box at the top left corner of the window and *RTT Start* button should be clicked. User can end RTT measurement by clicking the *RTT Stop* button. Measurement results returned from VoIP device are displayed to the *Sequence #* and *Network Delay* boxes simultaneously and also written to the log file.

When *RTT Start* button is clicked, Client A constructs determined sized UDP packets with determined intervals and sends it to the Client B. First byte of the UDP packet, the packet type indicator byte (PTI), indicates that this packet is an RTT measurement packet. Following four bytes includes the timer value of the microprocessor at the UDP packet's construction instant. Let's call this timer value

as TA_s, send time of Client A. Following two bytes indicate the sequence number of the send packet. After that, toggling binary 1's and 0's are inserted to the packets data section until packet size reaches the RTT packet length. Sequence number is incremented by one for every transmitted packet.

When Client B receives RTT measurement packet, it immediately changes the packet type indicator (PTI) byte, to indicate that, this is an RTT measurement acknowledge packet and transmits packet back to Client A. When Client A receives acknowledge packet, it immediately records current value of the microprocessor timer. Let's call this timer value as TA_r, receive time of Client A. Then, it reads sequence number and departure time of the packet from the beginning of the packets data section, calculates RTT by subtracting TA_s from TA_r.

$$RTT = TA\_r - TA\_s$$

UDP packet exchange for RTT measurements between clients A and B is seen in Figure 4.9.

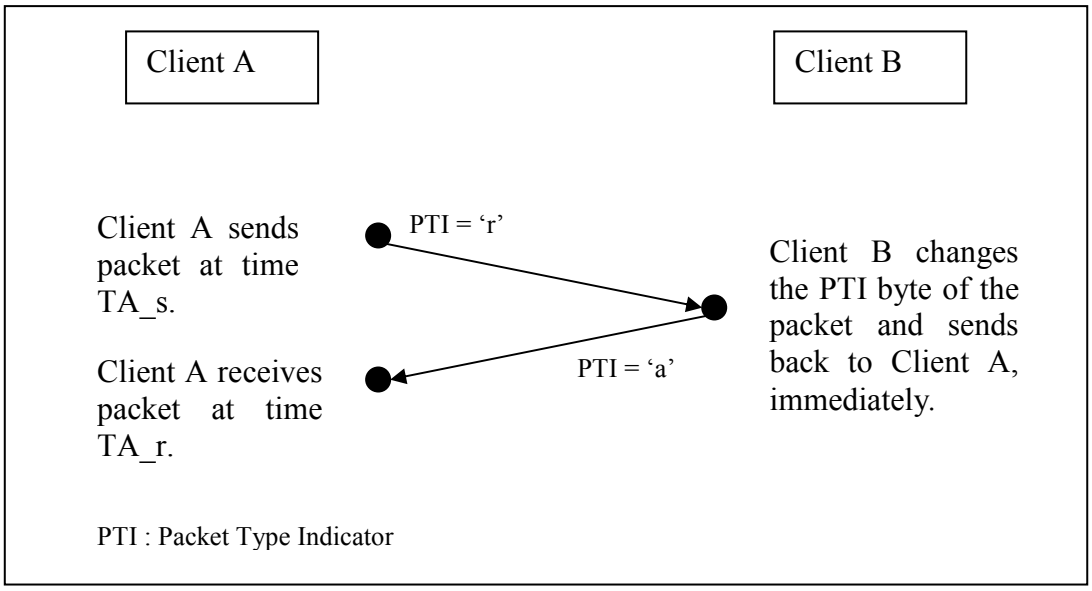

Figure 4.9 UDP packet exchange for RTT measurement between clients A and B

As mentioned, RTT is measured with respect to clock of Client A only, thus, there is no synchronization problem in RTT measurements. So, RTT measurements give more accurate results than one-way network delay measurements. Calculated RTT and corresponding sequence number is send to Serial User Interface via UART, in order to be displayed and recorded. RTT measurement continues until the *RTT stop* button is clicked.

We have measured RTT on two paths. For the first path, measurements on 28-29 June 2003, we have measured RTT on the path METU-HU. For the second path, measurements at 27 October to 14 November, we have measured RTT on the path GIT-METU. An example of observed RTT on the second path at 03:25 PM on 14 November 2003 is plotted at Figure 4.10.

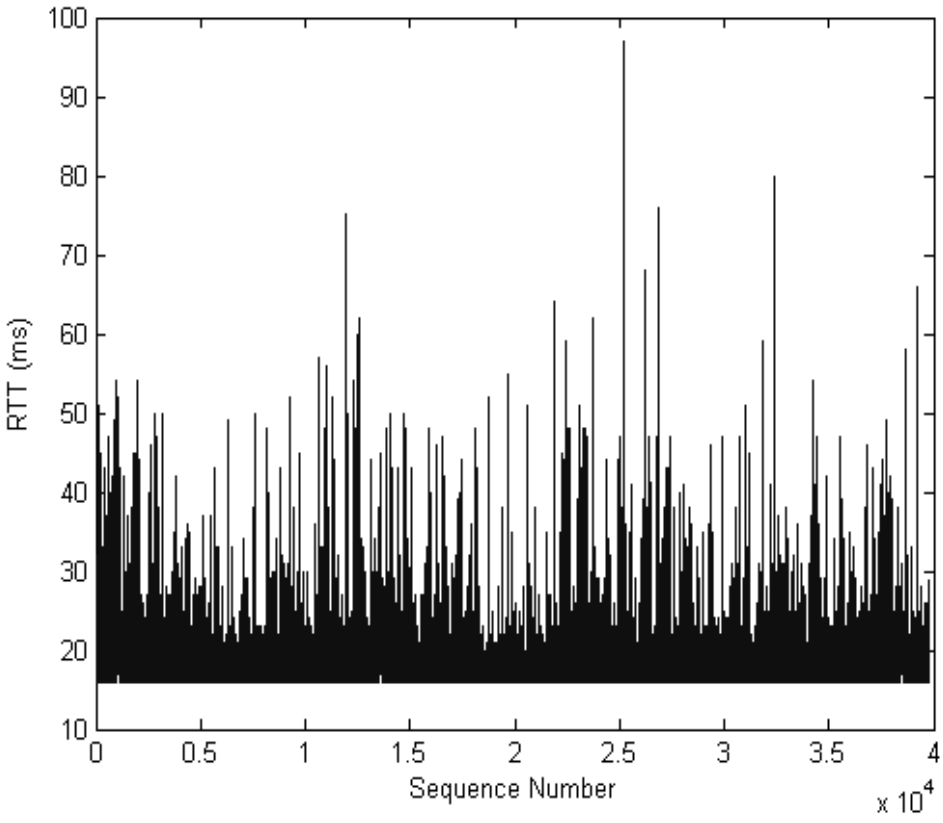

Figure 4.10 Example of measured RTT on the path GIT-METU

4.4 Packet Loss

Both loss and delay result from buffering within the network. As packets traverse the network, they are queued in buffers (thus adding to their end-end delay) and from time to time are dropped due to buffer overflow. As we mentioned before, we have measured one-way network delay and RTT in the Internet. We have used packet numbers in order to identify the delays experienced by each packet. We can also determine the packet losses using these packet numbers. Losses in this work is only the network losses, losses caused by late arrivals in play-out buffering applications are not included.

CHAPTER 5

DATA ANALYSIS AND RESULTS

5.1 Introduction

In this thesis, as we mentioned before, we have taken one-way network delay and RTT measurements on two paths. On the path METU-HU, we have taken 8 one-way network delay and 15 RTT measurements. On the path GIT-METU, we have taken 37 one-way network delay and 34 RTT measurements at working hours on workdays. We have attempted to find out the distribution of one-way network delay and RTT on two paths using these measurements. While we were analyzing the data observed from measurements, we have observed that, one-way network delay and RTT measurements taken on the path METU-HU are disturbed by the firewall of METU. We provided the change of firewall rules and then measured one-way network delay and RTT on the path GIT-METU.

We have extracted the normalized histograms of one-way network delay and RTT observations. Then we have obtained the maximum likelihood estimates of the parameters of gamma, lognormal and Weibull distributions for each normalized histogram, using statistics toolbox of MATLAB. Next, we have applied Kolmogorov-Smirnov goodness-of-fit test to the histograms and the estimated distribution functions. Then we have decided the observed distribution resembles to which assumed distribution using the outputs of Kolmogorov-Smirnov test. We have used functions of statistics toolbox of MATLAB for Kolmogorov-Smirnov goodness-of-fit test.

5.2 Assumed Distribution Functions

In this thesis, we attempted to fit observed one-way network delay and RTT distributions to gamma, lognormal and Weibull distributions. These distributions are explained in the following subsections.

5.2.1 Gamma Probability Density Function

The gamma distribution is a family of curves based on two parameters. The chi-square (X^2) and exponential distributions, which are children of the gamma distribution, are one-parameter distributions that fix one of the two gamma parameters.

The gamma pdf is,

$$y = f(x \mid \alpha, \beta) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)} x^{\alpha-1} e^{\frac{x}{\beta}}$$

Where, the gamma function $\Gamma(\alpha)$, is defined by the integral:

$$\Gamma(\alpha) = \int_0^{\infty} e^{-t} t^{\alpha-1} dt$$

The gamma function interpolates the factorial function. For integer n,

$$\Gamma(n+1) = n!$$

The gamma probability density function is useful in reliability models of lifetimes. When α is large, the gamma distribution closely approximates a normal distribution

with the advantage that the gamma distribution has density only for positive real numbers. Gamma distribution with parameters α=17, β=1.5 is shown in Figure 5.1.

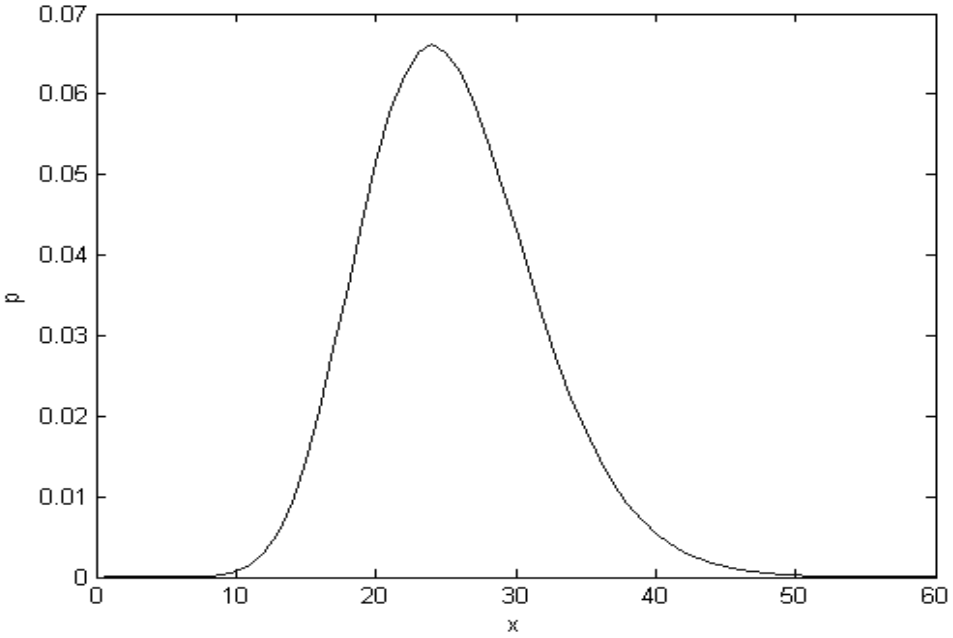

Figure 5.1 Example of a gamma distribution

5.2.2 Lognormal Probability Density Function

The lognormal pdf is,

$$y = f(x \mid \mu, \sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{\frac{-(\ln x - \mu)^2}{2\sigma^2}}$$

The normal and lognormal distributions are closely related. If X is distributed lognormal with parameters μ and σ^2, then $ln(X)$ is distributed normal with parameters μ and σ^2. The lognormal distribution is applicable when the quantity of interest must be positive, since $ln(X)$ exists only when the random variable X is

positive. Economists often model the distribution of income using a lognormal distribution.

An example of lognormal distribution with parameters, $\mu = 0$ and $\sigma = 1$ is shown in Figure 5.2.

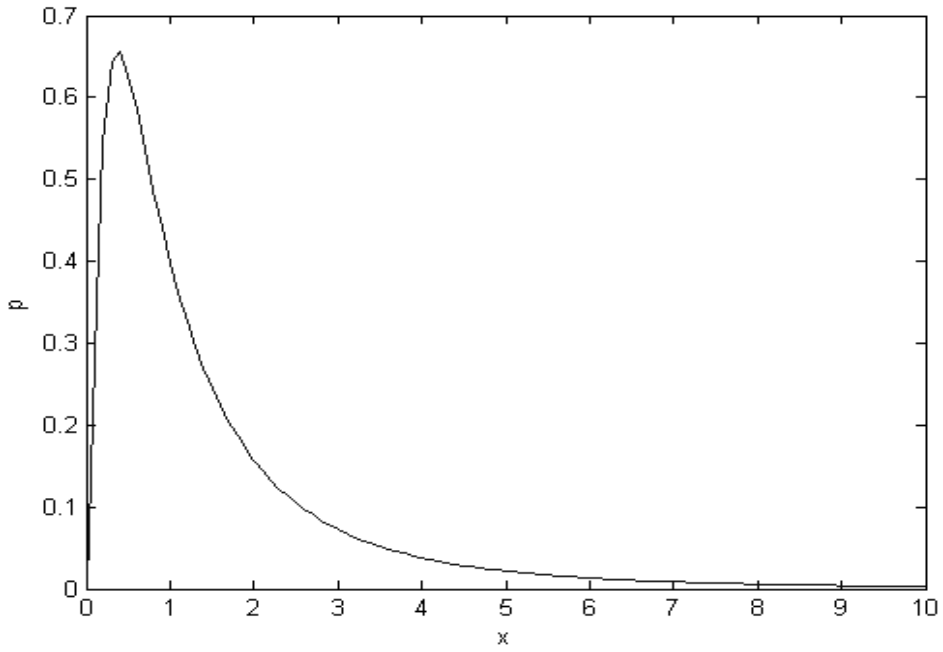

Figure 5.2 Example of a lognormal distribution

5.2.3 Weibull Probability Density Function

Waloddi Weibull (1939) offered the distribution that bears his name as an appropriate analytical tool for modeling the breaking strength of materials. Current usage also includes reliability and lifetime modeling. The Weibull distribution is more flexible than the exponential for these purposes.

The Weibull pdf is,

$$y = f(x \mid \alpha, \beta) = \alpha\beta x^{\beta-1} e^{-\alpha x^{\beta}} I_{(0,\infty)}(x)$$

An example of Weibull distribution with parameters, α = 5.8x10^{-8} and β = 5 is shown in Figure 5.3.

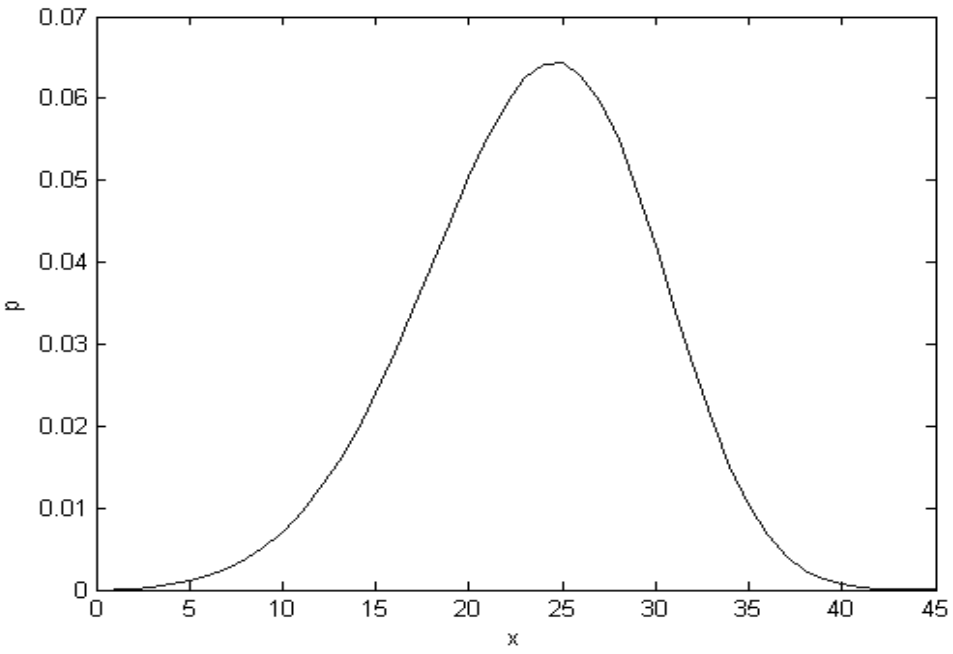

Figure 5.3 Example of a Weibull distribution

5.3 Kolmogorov-Smirnov Goodness-of-fit Test

The object of this test is to investigate the significance of the difference between an observed distribution and a specified population distribution. The cumulative distribution S(x) is determined from the observed distribution. The cumulative distribution F(x) of the assumed distribution is also determined. The maximum difference between these two distributions, D = | F-S |, provides the test statistics and this value is compared with the value D(α) obtained from Table 5.1. If D>D(α) it is rejected that the observed distribution comes from the assumed distribution. We have used "KSTEST" function of the statistics toolbox of MATLAB for this test.

This function returns its output variable, H, equals 0 if observed distribution matches to assumed distribution, else 1. KSTEST function also gives D and D(α) values. We have used these D values for deciding that the observed distribution resembles to which distribution.

Table 5.1 Critical values of D for the Kolmogorov-Smirnov goodness-of-fit test

n	Level of Significance α				
	0.20	0.15	0.10	0.05	0.01
1	0.900	0.925	0.950	0.975	0.993
2	0.684	0.726	0.776	0.842	0.929
3	0.565	0.597	0.642	0.708	0.823
4	0.494	0.525	0.564	0.624	0.733
5	0.446	0.474	0.510	0.565	0.669
6	0.410	0.436	0.470	0.521	0.618
7	0.381	0.405	0.438	0.486	0.577
8	0.358	0.381	0.411	0.457	0.543
9	0.339	0.360	0.388	0.432	0.514
10	0.322	0.342	0.368	0.410	0.490
11	0.307	0.326	0.352	0.391	0.468
12	0.295	0.313	0.338	0.375	0.450
13	0.284	0.302	0.325	0.361	0.433
14	0.274	0.292	0.314	0.349	0.418
15	0.266	0.283	0.304	0.338	0.404
16	0.258	0.274	0.295	0.328	0.392
17	0.250	0.266	0.286	0.318	0.381
18	0.244	0.259	0.278	0.309	0.371
19	0.237	0.252	0.272	0.301	0.363
20	0.231	0.246	0.264	0.294	0.356
25	0.210	0.220	0.240	0.270	0.320
30	0.190	0.200	0.220	0.240	0.290
35	0.180	0.190	0.210	0.230	0.270
Over 35	$\frac{1.07}{\sqrt{n}}$	$\frac{1.14}{\sqrt{n}}$	$\frac{1.22}{\sqrt{n}}$	$\frac{1.36}{\sqrt{n}}$	$\frac{1.63}{\sqrt{n}}$

Figure 5.4 shows the CDF of observed one-way network at 10:10 AM at 30 October 2003. CDF of assumed gamma, lognormal and Weibull distributions are also shown.

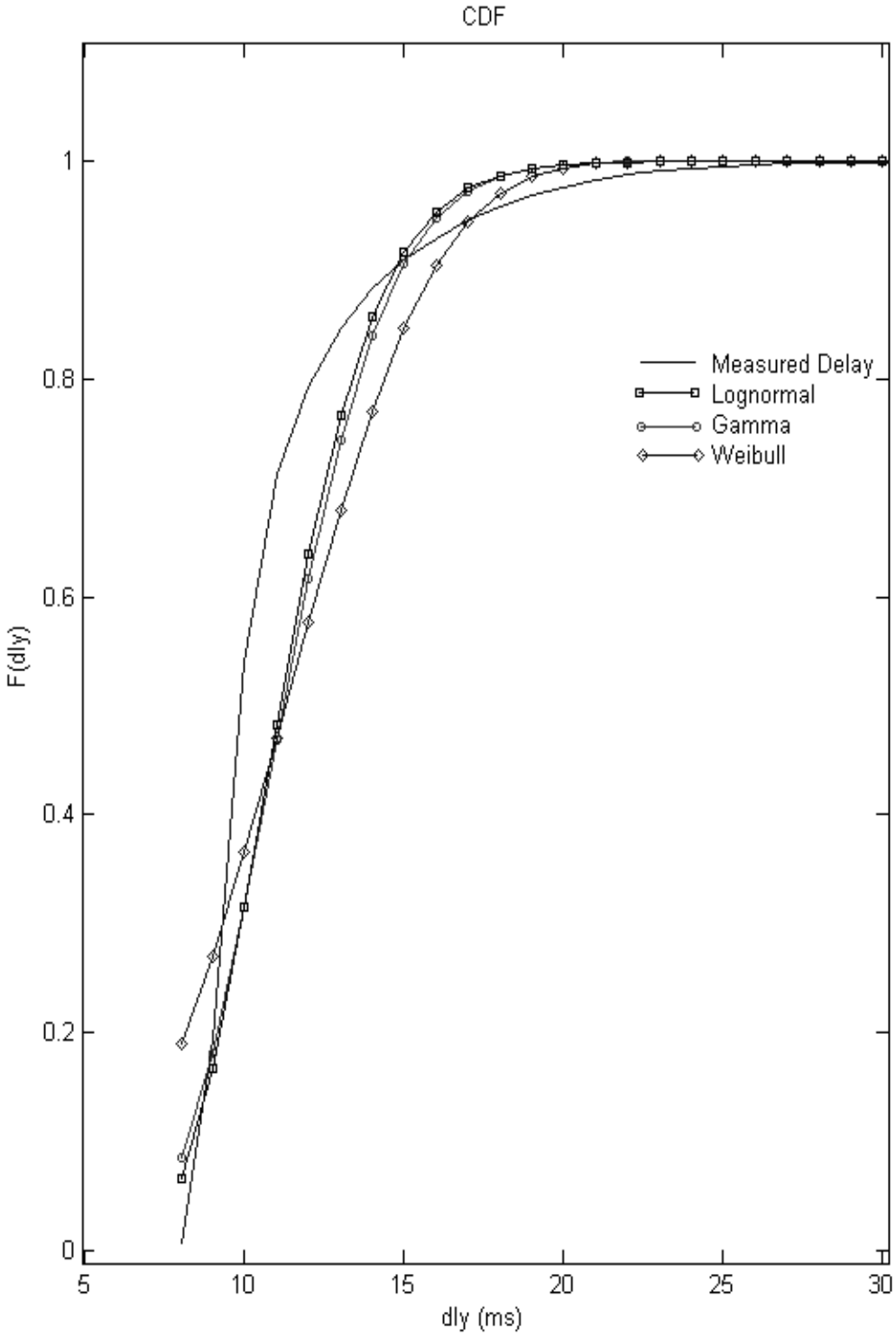

Figure 5.4 CDF plot of observed one-way network delay and assumed distributions

5.4 Effect of Firewall on Measurements

We have measured one-way network delay, RTT distribution and packet loss rate at the path METU-HU. We have observed high delay values and loss rates although HU and METU are not located apart. Then, we have located a VoIP device at LAN of GIT and take one-way network delay measurements. We have observed that delay distribution is similar to the path METU-HU. Observed distribution is shown in Figure 5.5. Then, we have investigated the firewall rules of METU. VoIP devices used in our measurements communicate on port 1024. We have provided necessary rule changes at the firewall of METU so, incoming packets on port 1024 are not processed by the firewall of METU. Then we have measured one-way network delay from GIT to METU again, and observed that distribution of one-way network delay has been changed. One-way network delay and RTT distributions on the path GIT-METU, after firewall rule changes, are similar to the results of Test Traffic Measurements (TTM) of Réseaux IP Européens Network Coordination Center (RIPE NCC). An example of the distribution of one-way network delay after firewall rule changes was shown in Figure 5.6. Figures 5.5 and 5.6 clearly show the effect of firewall on our measurements.

We have also observed that packet processing at the firewall dramatically increases packet loss rate. We have observed a packet loss rate of about 14 % at one-way network delay measurement from GIT to METU, before the rules of the firewall at METU have been changed and observed less than 1 % packet loss rate after the rule change.

Our measurements on the path METU-HU are affected by the packet processing at the firewall of METU. Measurements on the path GIT-METU are taken after the rules of the firewall at METU have been changed, so, measurements on the path GIT-METU are not affected by the packet processing at the firewall of METU.

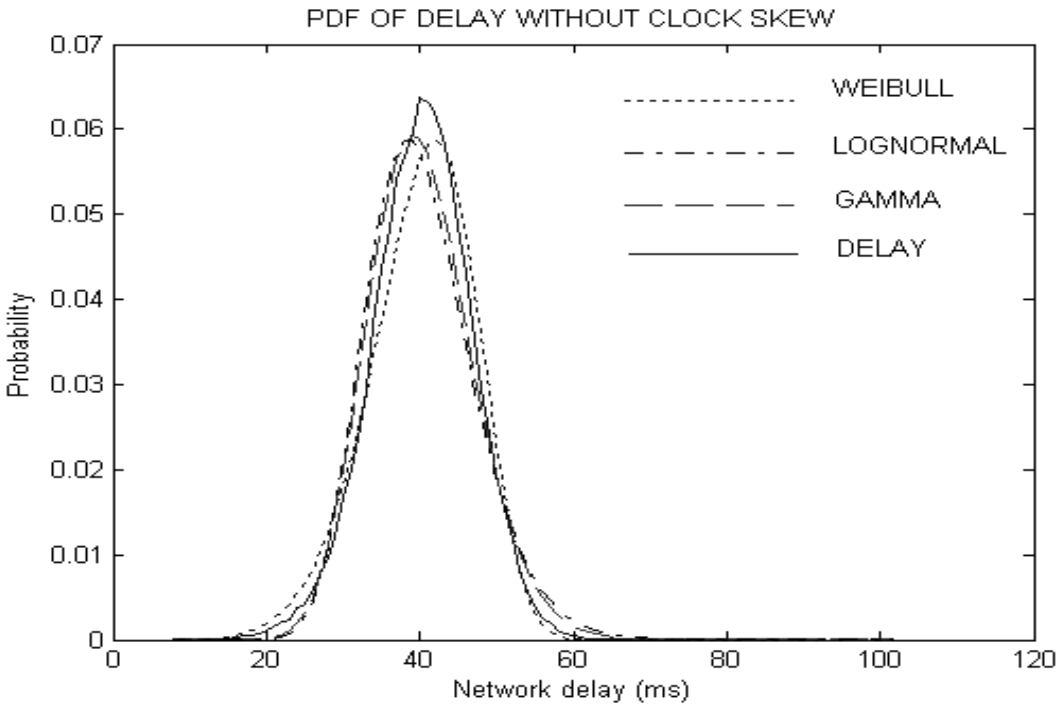

Figure 5.5 PDF of one-way network delay from GIT to METU before rule change

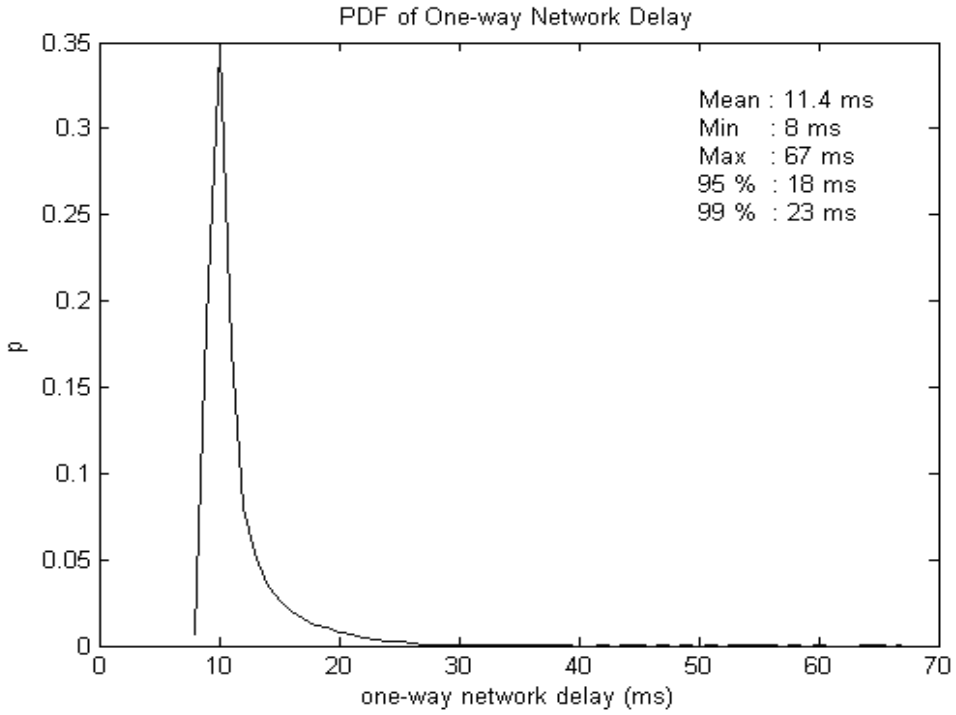

Figure 5.6 PDF of one-way network delay from GIT to METU after rule change

5.5 Measurements and Data Analysis

VoIP devices are located to the LANs of Middle East Technical University (METU), Hacettepe University (HU) and Gebze Institute of Technology (GIT) for one-way network delay, Round Trip Time (RTT) and packet loss rate measurements. So, measurements are achieved on two paths, METU-HU and GIT-METU. For one-way delay measurements, fixed size IP probe packets of 135 bytes are send with a period of 16 ms. Probe packets consist of 7 bytes of modified RTP header and 128 bytes of digitized voice data corresponding to 16 ms of voice signal. For RTT measurements, fixed size IP probe packets of 128 bytes are send with a period of 50 ms. Probe packets consist of 7 bytes of modified RTP header and 121 bytes of random generated data to fill the packet. These packet length and packet generation period values are chosen to extract distributions of one-way network delay and RTT successively, without loading the network with probe packets.

UDP which is not a connection-oriented protocol is used in the VoIP device. The master device, which controls the measurements, starts and ends the measurement without notifying the slave device and collects the data. When it ends a measurement, it behaves to the last arrived packet before it ends the communication as the last measurement packet. Some of the packet sent before this packet may arrive to the master after this packet if they had experienced higher delay and they are regarded as lost packets. This situation does not affect our statistics considering the number of packets used in a measurement.

Measurements on path METU-HU are controlled by the VoIP device located at HU, so measured one-way network delay is from METU to HU. 8 one-way network delay and 15 RTT measurements are taken on this path at working hours on 28-29 June 2003. About 240.000 packets are sent for one-way network delay measurements and 280.000 packets are sent for RTT measurements. Measurements are effected by the processing at the firewall of METU. We have provided

necessary rule changes at the firewall of METU, so our packets are not processed at the firewall.

Measurements on path GIT-METU are controlled by the VoIP device located at METU, so measured one-way network delay is from GIT to METU. 37 one-way network delay and 34 RTT measurements are taken on this path at working hours at work days from 27 October to 14 November 2003. About 2.5 million packets are sent for one-way network delay measurements and 2.3 million packets are sent for RTT measurements.

5.5.1 Measurements on the path METU-HU

After we have collected the one-way network delay and RTT data on path METU-HU, we have analyzed data using statistics toolbox of MATLAB. Moon's algorithm is used for elimination of clock skew effect on measured one-way network delays. Then, normalized histograms of obtained one-way network delay and RTT measurements are extracted. Next, maximum likelihood estimates of the parameters of gamma, lognormal and Weibull distributions for each normalized histogram are obtained. Kolmogorov-Smirnov goodness-of-fit test is applied to the histograms and assumed distributions. Decision of the observed distribution resembles to which assumed distribution is made using the outputs of Kolmogorov-Smirnov test.

5.5.1.1 One-way Network Delay from METU to HU

We have taken 8 one-way network delay measurements on the path METU-HU. Measurements are summarized at Table 5.2. An example of one-way network delay from METU to HU after skew elimination is shown in Figure 5.7.

Table 5.2. One-way network delay measurements summary (METU-HU)

Time of Day	# Send Packets	# Received Packets	# Lost Packets	% Loss	Mean Delay (ms)	Std. Dev. Delay	Min Delay (ms)	Max Delay (ms)
10:40	5500	5293	207	3.76	30.39	7.5	6	58
11:05	13446	12444	1002	7.45	31.67	6.46	7	56
11:10	34962	31875	3087	8.83	26.55	6.02	4	142
11:45	73950	66542	7408	10.02	30.45	5.77	6	81
13:00	92322	84205	8117	8.79	37.88	6.99	11	87
13:25	4243	3768	475	11.19	30.53	6.3	4	73
14:20	4716	4223	493	10.45	30.73	6.09	11	70
14:40	10039	9079	960	9.56	30.09	5.89	8	51

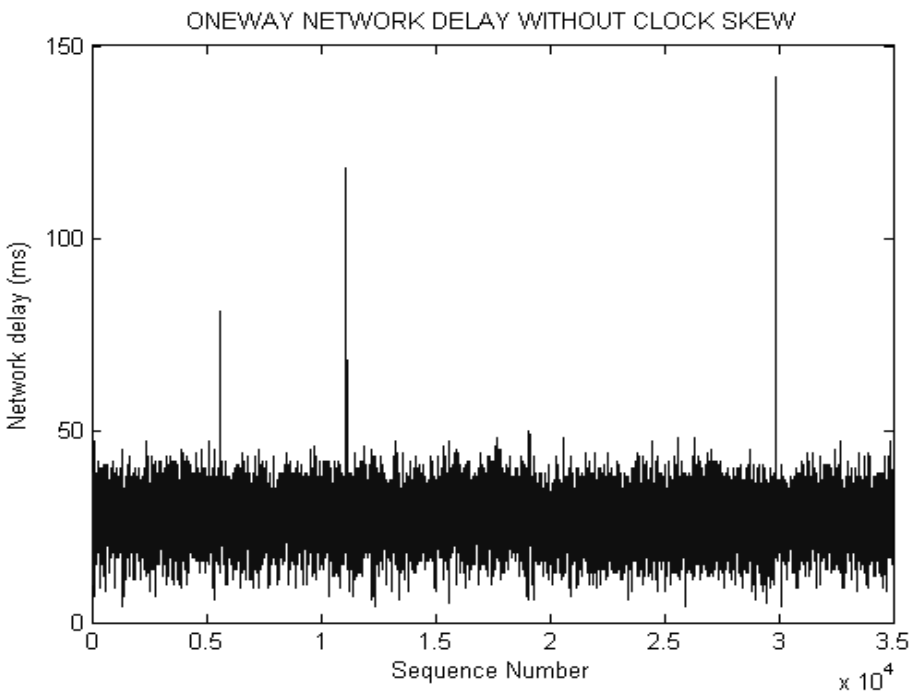

Figure 5.7 One-way network delay from METU to HU (11:10 AM)

Normalized histogram of the one-way network delay shown in Figure 5.7 is plotted in Figure 5.8. Assumed gamma, lognormal and Weibull distributions for the observed one-way network delay are also shown in Figure 5.8. Kolmogorov-Smirnov Goodness-of-fit test results for one-way delay measurements of the path METU-HU are summarized in Table 5.3. We have observed that, for all one-way

network delay observations, assumed Weibull distributions returned less D values then assumed gamma or lognormal distributions. We concluded that one-way network delay values from METU to HU can be modeled with Weibull distribution with different parameters at different time of the day.

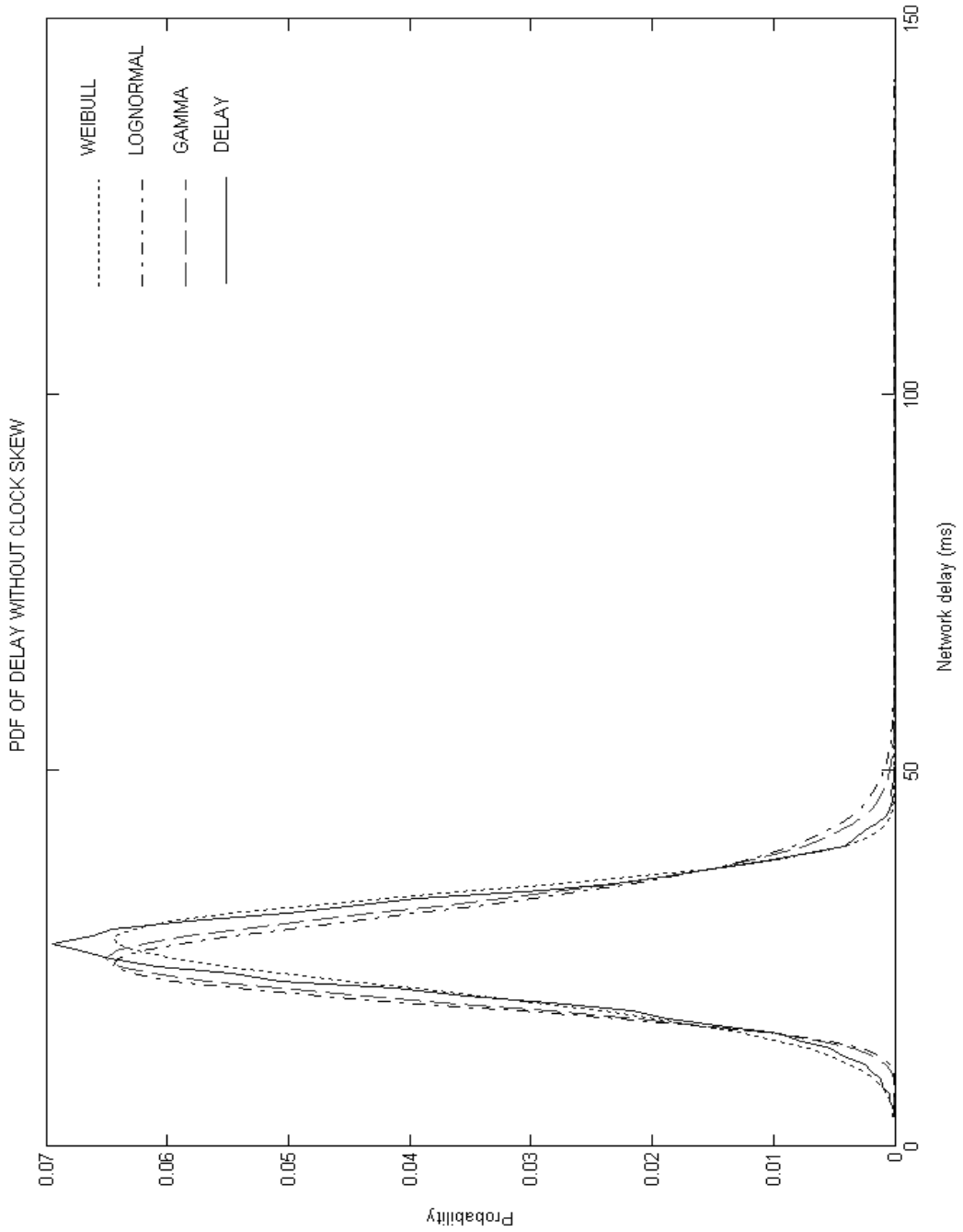

Figure 5.8 PDF of one-way network delay from METU to HU (11:10 AM)

Table 5.3. Kolmogorov-Smirnov goodness-of-fit test results for one-way network delay measurements from METU to HU

Time of Day	Gamma		Lognormal		Weibull		D			D(α)
	α	β	μ	σ	α	β	Gamma	Logn	Weibull	
10:40	13.8088	2.2006	3.3774	0.2886	0	4.6215	0.0858	0.1066	**0.0354**	0.0186
11:05	22.0505	1.4362	3.4325	0.2208	0	5.4433	0.0732	0.0874	**0.0532**	0.0121
11:10	17.8446	1.4881	3.2509	0.2470	0	4.9538	0.0729	0.0899	**0.0511**	0.0076
11:45	25.8689	1.1771	3.3966	0.2028	0	5.8556	0.0693	0.0823	**0.0547**	0.0053
13:00	27.9211	1.3569	3.6166	0.1936	0	5.9132	0.0556	0.0684	**0.0554**	0.0047
13:25	21.3379	1.4306	3.3950	0.2256	0	5.4636	0.0709	0.0860	**0.0483**	0.0221
14:20	23.9408	1.2838	3.4044	0.2103	0	5.5096	0.0703	0.0840	**0.0542**	0.0208
14:40	24.2146	1.2425	3.3833	0.2097	0	5.6919	0.0823	0.0949	**0.0595**	0.0142

5.5.1.2 Round Trip Time (RTT) on the path METU-HU

We have taken 15 one-way network delay measurements on the path METU-HU. Measurements are summarized at Table 5.4.

Table 5.4. RTT measurements summary (METU-HU)

Time of Day	# Send Packets	# Received Packets	# Lost Packets	% Loss	Mean Delay (ms)	Std. Dev. Delay (ms)	Min Delay (ms)	Max Delay (ms)
11:30	58960	49728	9232	15.66	51.69	9.60	5	92
12:15	2031	1812	219	10.78	45.86	12.8	4	89
12:20	52848	45952	6896	13.05	49.13	10.47	3	100
12:45	8246	7015	1231	14.93	50.69	9.61	6	83
13:15	12879	10837	2042	15.85	59.98	9.34	15	115
13:25	2172	1808	364	16.76	58.24	9.21	12	81
13:30	3481	2886	595	17.09	59.60	9.32	12	84
13:40	5909	4944	965	16.33	59.34	9.81	7	86
13:45	4627	3879	748	16.17	59.08	9.78	8	99
14:30	15374	12882	2492	16.21	54.63	9.66	4	107
14:40	30639	25825	4814	15.71	54.19	9.24	5	101
14:50	4653	3867	786	16.89	56.90	11.12	8	115
15:03	2616	2192	424	16.21	48.95	10.05	5	99
15:30	69430	60254	9176	13.22	51.26	9.96	3	107
16:00	6214	5296	918	14.77	53.33	10.70	6	98

An example of RTT on the path METU-HU is shown in Figure 5.9. Normalized histogram of the RTT shown in Figure 5.9 is plotted in Figure 5.10. Assumed gamma, lognormal and Weibull distributions for the observed RTT are also shown in Figure 5.10. Kolmogorov-Smirnov Goodness-of-fit test results for RTT measurements of the path METU-HU are summarized in Table 5.5. We have observed that, for most of the RTT observations, assumed Weibull distributions returned less D values then assumed gamma or lognormal distributions. We concluded that RTT values on the path METU-HU can be modeled with Weibull distribution with different parameters at different time of the day.

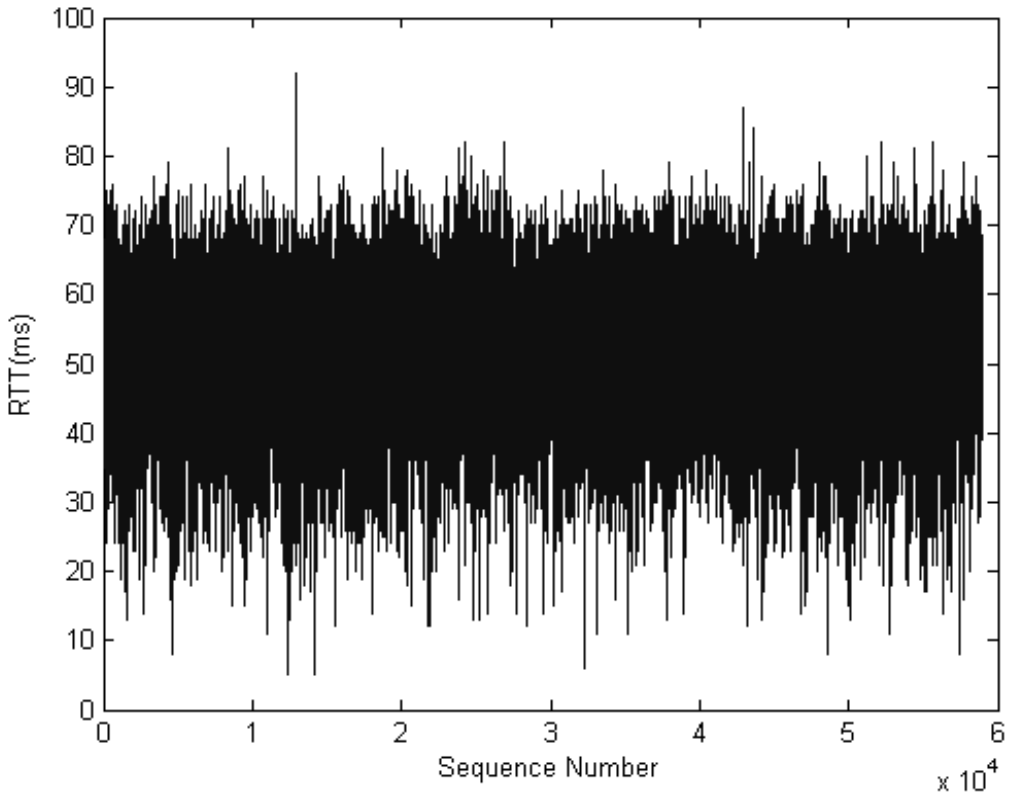

Figure 5.9 RTT on the path METU to HU (11:30 AM)

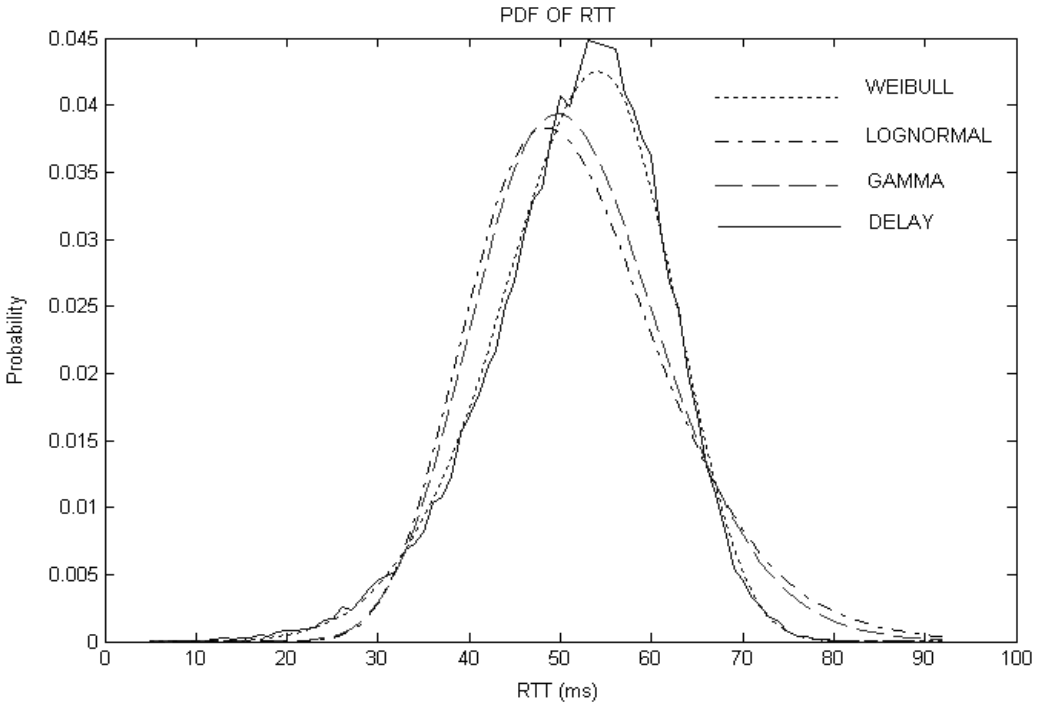

Figure 5.10 PDF of RTT on the path METU-HU (11:30 AM)

Table 5.5. Kolmogorov-Smirnov goodness-of-fit test results for RTT measurements on the path METU-HU

Time of Day	Gamma		Lognormal		Weibull		D			D(α)
	α	β	μ	σ	α	β	Gamma	Logn	Weibull	
11:30	25.1750	2.0532	3.9253	0.2098	0	6.3384	0.0887	0.1030	**0.0303**	0.0061
12:15	10.0008	4.5858	3.7748	0.3497	0	4.1753	0.1055	0.1255	**0.0557**	0.0318
12:20	17.9244	2.7408	3.8663	0.2554	0	5.5722	0.0997	0.1171	**0.0399**	0.0063
12:45	23.7772	2.1319	3.9046	0.2174	0	6.2181	0.0897	0.1046	**0.0305**	0.0162
13:15	36.9657	1.6226	4.0804	0.1707	0	7.6609	0.0834	0.0947	**0.0262**	0.0130
13:25	35.0021	1.6638	4.0502	0.1771	0	7.5149	0.0953	0.1085	**0.0395**	0.0318
13:30	36.0658	1.6527	4.0738	0.1739	0	7.6136	0.0913	0.1035	**0.0312**	0.0252
13:40	31.9016	1.8602	4.0676	0.1861	0	7.1912	0.0916	0.1034	**0.0333**	0.0193
13:45	31.8078	1.8574	4.0631	0.1865	0	7.0348	0.0879	0.0999	**0.0371**	0.0218
14:30	27.8574	1.9611	3.9825	0.1992	0	6.7201	0.0927	0.1056	**0.0346**	0.0119
14:40	29.9988	1.8063	3.9757	0.1917	0	6.9637	0.0936	0.1063	**0.0339**	0.0084
14:50	24.3472	2.3372	4.0207	0.2109	0	5.1564	**0.0785**	0.0935	0.0877	0.0218
15:03	19.3869	2.5247	3.8647	0.2452	0	5.5340	0.1188	0.1307	**0.0699**	0.0289
15:30	22.4285	2.2855	3.9145	0.2247	0	6.0498	0.0900	0.1061	**0.0315**	0.0055
16:00	20.5901	2.5902	3.9521	0.2364	0	5.9049	0.0986	0.1136	**0.0392**	0.0186

5.5.2 Measurements on the path GIT-METU

After we have collected the one-way network delay and RTT data on path GIT-METU, we have analyzed data using statistics toolbox of MATLAB. Moon's algorithm is used for elimination of clock skew effect on measured one-way network delays. Then, normalized histograms of obtained one-way network delay and RTT measurements are extracted. Next, maximum likelihood estimates of the parameters of gamma, lognormal and Weibull distributions for each normalized histogram are obtained. Kolmogorov-Smirnov goodness-of-fit test is applied to the histograms and assumed distributions. Decision of the observed distribution resembles to which assumed distribution is made using the outputs of Kolmogorov-Smirnov test.

5.5.2.1 One-way Network Delay from GIT to METU

We have taken 37 one-way network delay measurements on the path GIT-METU. Measurements are summarized at Table 5.6. An example of one-way network delay from GIT to METU after skew elimination is shown in Figure 5.11.

Table 5.6. One-way network delay measurements summary (GIT-METU)

Time of Day	# Send Packets	# Received Packets	# Lost Packets	% Loss	Mean Delay (ms)	Std. Dev. Delay (ms)	Min Delay (ms)	Max Delay (ms)
08:45	65608	65568	40	0.06	11.22	2.05	9	58
09.05	69817	69800	17	0.04	10.90	1.34	9	55
10:05	23540	23533	7	0.03	9.07	1.45	7	52
10:10	67285	67116	169	0.25	11.40	3.06	8	67
10:15	75851	75800	51	0.07	12.09	1.98	9	50
10:25	34394	34352	42	0.12	12.42	2.09	10	77
10:35	68151	68034	117	0.17	11.19	2.49	8	76
10:45	79708	79680	28	0.03	9.83	1.67	8	45
11:15	71354	71226	128	0.18	12.13	2.77	9	64
11:50	56103	56074	29	0.05	9.87	1.52	8	69
12:00	77177	77002	175	0.23	11.94	3.00	8	116
12:00	112889	112746	143	0.13	10.04	2.03	8	135
12:00	79632	79594	38	0.05	9.37	1.75	7	52
12:00	52778	52711	67	0.13	9.90	2.45	7	75
12:20	38699	38646	53	0.14	9.61	2.09	7	80
12:55	95016	94367	649	0.68	8.56	1.72	7	62
13:00	10637	10591	46	0.43	13.66	3.45	11	125
13:05	19745	19723	22	0.11	8.52	1.90	7	92
13:15	70969	70842	127	0.18	9.60	2.27	8	127
13:15	41560	41546	14	0.03	8.88	1.29	7	59
13:20	84930	84711	219	0.26	10.80	2.95	8	122
13:30	33223	33087	136	0.41	12.81	3.65	9	97
13:55	93030	93004	26	0.03	9.89	1.10	8	99
14:00	66304	66186	118	0.18	8.13	2.19	6	126
14:00	15289	15288	1	0.01	6.81	1.36	5	39
14:10	77302	77096	206	0.27	9.16	3.08	7	159
14:35	92656	92537	119	0.13	9.64	2.17	7	111
14:50	91967	91534	433	0.47	12.15	4.02	8	161
15:00	72875	72853	22	0.03	8.55	1.32	7	47
15:00	197535	197387	148	0.07	10.35	1.91	8	90
15:05	71554	71402	152	0.21	9.72	2.25	7	66
15:15	80558	80505	53	0.07	10.24	1.63	8	54
15:30	81376	81124	252	0.31	10.25	4.57	7	517
15:45	20034	19997	37	0.18	11.03	2.87	8	117
16:00	76629	76257	372	0.48	9.28	3.63	6	70
16:00	62664	61953	711	1.13	12.19	4.33	7	123
16:05	79527	79514	13	0.02	9.39	1.17	8	47

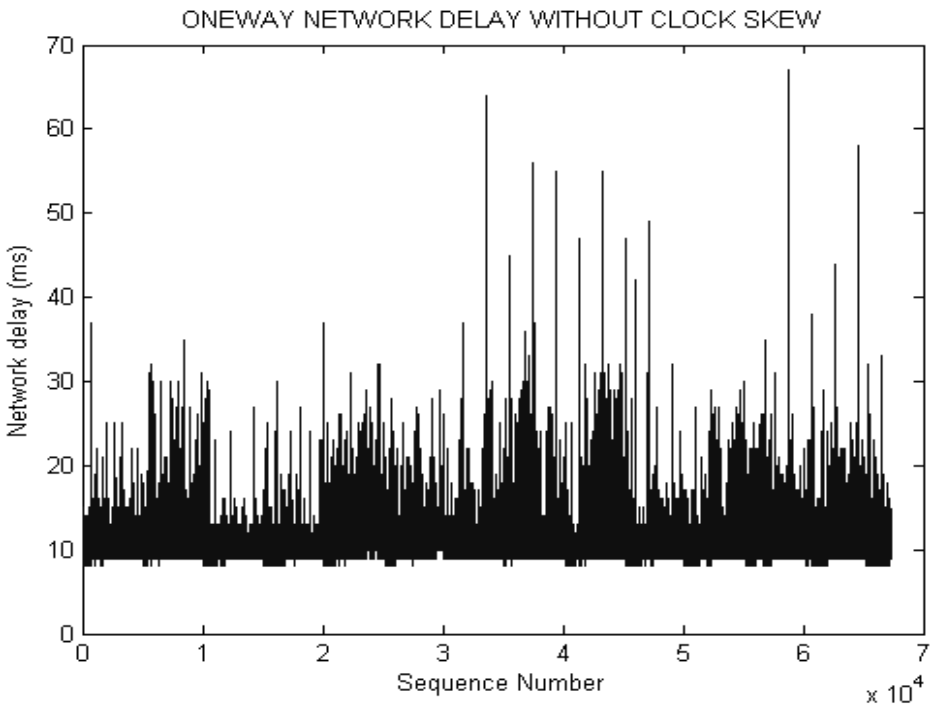

Figure 5.11 One-way network delay from GIT to METU (10:10 AM)

Normalized histogram of the one-way network delay shown in Figure 5.11 is plotted in Figure 5.12. Normalized histogram of one-way network delay and assumed gamma, lognormal and Weibull distributions are shown in Figure 5.13. Kolmogorov-Smirnov Goodness-of-fit test results for one-way delay measurements of the path METU-HU are summarized in Table 5.7. We have observed that, for most of the one-way network delay observations, assumed lognormal distributions returned less D values then assumed gamma or Weibull distributions. We concluded that one-way network delay values from GIT to METU can be modeled with lognormal distribution with different parameters at different time of the day.

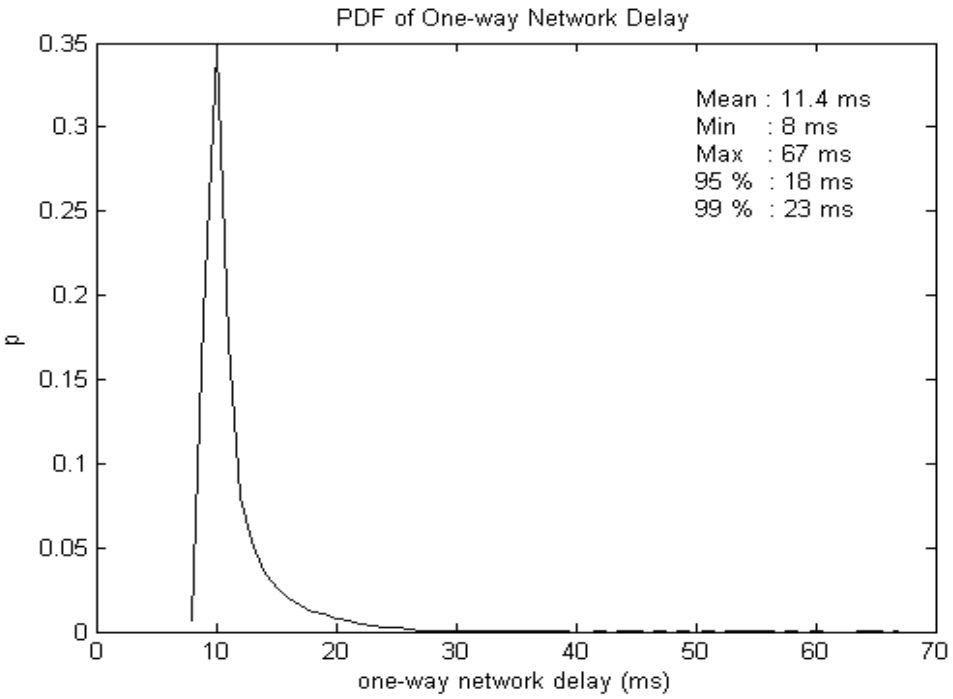

Figure 5.12 PDF of one-way network delay from GIT to METU (10:10 AM)

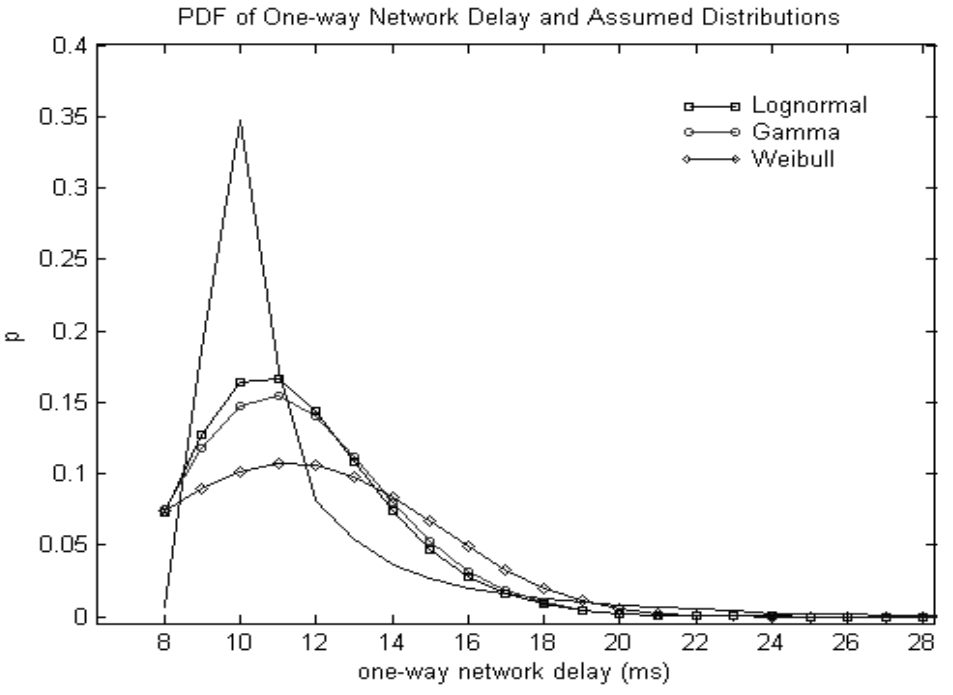

Figure 5.13 PDF of one-way network delay and assumed distributions (detailed)

Table 5.7. Kolmogorov-Smirnov goodness-of-fit test results for one-way network delay measurements from GIT to METU

Time of Day	Gamma		Lognormal		Weibull		D			D(α)
	α	β	μ	σ	α	β	Gamma	Logn	Weib	
08:45	43.0334	0.2608	2.4063	0.1414	0.0000	4.6011	0.3117	**0.3034**	0.3379	0.0053
09.05	86.5886	0.1259	2.3833	0.1022	0.0000	8.1492	0.2855	**0.2756**	0.2943	0.0051
10:05	55.2475	0.1642	2.1959	0.1262	0.0000	5.6348	0.3265	**0.3172**	0.3177	0.0088
10:10	18.8361	0.6055	2.4073	0.2171	0.0001	3.4843	0.2440	**0.2298**	0.2639	0.0052
10:15	47.8056	0.2529	2.4817	0.1372	0.0000	5.0061	0.2745	**0.2643**	0.2983	0.0049
10:25	49.9005	0.2488	2.5090	0.1331	0.0000	5.4866	0.2651	**0.2606**	0.2866	0.0073
10:35	27.1100	0.4126	2.3961	0.1811	0.0001	3.9474	0.2567	**0.2436**	0.2626	0.0052
10:45	47.0220	0.2091	2.2752	0.1373	0.0000	5.4057	0.2641	**0.2500**	0.3003	0.0048
11:15	25.7292	0.4715	2.4762	0.1860	0.0000	3.9390	0.2403	**0.2399**	0.2797	0.0051
11:50	60.3893	0.1635	2.2818	0.1209	0.0000	6.4304	0.2808	**0.2682**	0.2917	0.0057
12:00	22.6270	0.5278	2.4578	0.1980	0.0000	3.9495	0.2434	**0.2387**	0.2563	0.0049
12:00	60.4254	0.1662	2.2988	0.1132	0.0000	8.1008	0.2952	**0.2850**	0.2945	0.0040
12:00	38.9460	0.2405	2.2243	0.1513	0.0000	5.0220	0.2710	**0.2651**	0.2702	0.0048
12:00	23.0298	0.4298	2.2704	0.1957	0.0001	3.7908	0.2538	**0.2519**	0.2707	0.0059
12:20	31.8674	0.3015	2.2469	0.1646	0.0000	4.2461	0.2770	**0.2754**	0.2814	0.0069
12:55	37.5001	0.2284	2.1342	0.1513	0.0000	4.8458	**0.2863**	0.2878	0.2927	0.0044
13:00	23.2628	0.5871	2.5927	0.1946	0.0000	4.1004	0.2427	**0.2331**	0.2538	0.0132
13:05	46.4532	0.1834	2.1314	0.1294	0.0000	7.3334	**0.2674**	0.2772	0.2872	0.0097
13:15	57.2692	0.1677	2.2534	0.1107	0.0000	8.6503	0.2793	**0.2559**	0.2943	0.0051
13:15	68.6511	0.1294	2.1770	0.1130	0.0000	8.0586	0.3105	**0.2981**	0.3121	0.0067
13:20	24.0965	0.4484	2.3591	0.1855	0.0001	3.8862	0.2721	**0.2713**	0.2972	0.0047
13:30	16.1700	0.7920	2.5187	0.2371	0.0001	3.4710	0.2038	**0.2008**	0.2366	0.0075
13:55	117.6079	0.0841	2.2872	0.0870	0.0000	10.6381	0.3052	**0.2953**	0.3139	0.0045
14:00	35.5696	0.2287	2.0820	0.1475	0.0000	5.6980	0.2955	**0.2852**	0.2893	0.0053
14:00	41.5089	0.1640	1.9059	0.1424	0.0000	6.2779	0.3079	**0.2891**	0.3107	0.0110
14:10	25.6221	0.3574	2.1948	0.1701	0.0000	4.5620	0.3031	**0.2891**	0.3038	0.0049
14:35	31.8134	0.3031	2.2505	0.1632	0.0000	4.4462	0.2494	**0.2345**	0.2698	0.0045
14:50	14.9932	0.8105	2.4637	0.2422	0.0001	3.5096	0.2077	**0.1998**	0.2178	0.0045
15:00	55.1787	0.1550	2.1369	0.1279	0.0000	6.0466	**0.2622**	0.2642	0.2911	0.0050
15:00	42.6710	0.2425	2.3250	0.1432	0.0000	5.3986	0.2684	**0.2628**	0.2630	0.0031
15:05	26.6818	0.3641	2.2549	0.1812	0.0001	4.0614	0.2675	**0.2648**	0.2743	0.0051
15:15	57.7890	0.1772	2.3177	0.1221	0.0000	7.3220	0.3108	0.3059	**0.2898**	0.0048
15:30	19.8568	0.5164	2.3022	0.1974	0.0001	3.9297	0.2524	**0.2373**	0.2493	0.0048
15:45	26.2360	0.4204	2.3813	0.1791	0.0000	4.4094	**0.2317**	0.2350	0.2544	0.0096
16:00	9.7850	0.9486	2.1761	0.2962	0.0025	2.5568	0.2719	**0.2647**	0.2839	0.0049
16:00	9.7921	1.2451	2.4489	0.3114	0.0004	2.9839	0.1636	**0.1557**	0.1691	0.0055
16:05	91.1905	0.1029	2.2337	0.0984	0.0000	9.4092	0.2963	0.2991	**0.2923**	0.0048

We have observed that all of the observed one-way delay distributions fall into Class A distributions introduced by [17]. An example of normalized distribution of one-way network delay is shown in Figure 5.14.

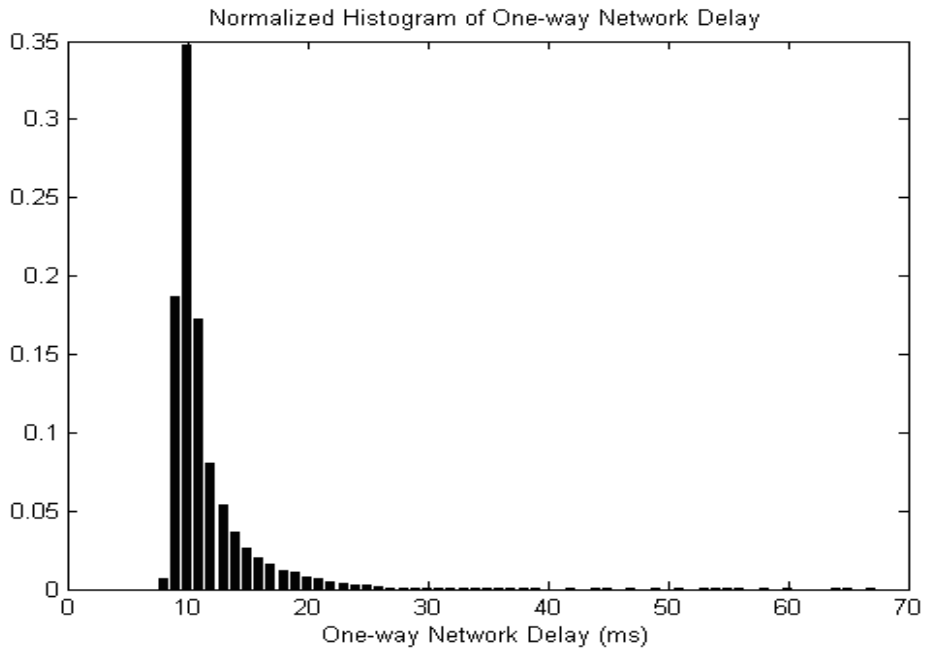

Figure 5.14 Example of normalized histogram of observed one-way network delay

5.5.2.2 Round Trip Time (RTT) on the path GIT-METU

We have taken 34 one-way network delay measurements on the path GIT-METU. Measurements are summarized at Table 5.8.

Table 5.8. RTT measurements summary (GIT-METU)

Time of Day	# Send Packets	# Received Packets	# Lost Packets	% Loss	Mean Delay (ms)	Std. Dev. Delay (ms)	Min Delay (ms)	Max Delay (ms)
09:15	69469	68733	736	1.06	18.49	3.02	16	127
09:25	50452	49865	587	1.16	18.54	3.48	16	182
09:40	58560	57752	808	1.38	18.18	2.76	16	80
10:35	46045	45488	557	1.21	19.09	4.04	16	66
10:40	86080	85074	1006	1.17	18.46	3.34	16	175
10:45	152620	150323	2297	1.50	18.86	3.64	16	72
11:00	57629	56808	821	1.42	20.79	5.16	16	103
11:10	49974	49169	805	1.61	18.65	3.75	16	330
11:15	55389	54533	856	1.54	19.28	4.95	16	135
11:35	19965	19780	185	0.93	18.13	2.88	16	76
12:15	28854	28419	435	1.51	17.54	1.99	16	71
12:20	13602	13413	189	1.39	17.78	2.81	16	187
12:20	11670	11547	123	1.05	18.84	3.57	16	72
12:25	7162	6909	253	3.53	27.89	8.95	16	71
12:30	110888	108707	2181	1.97	18.07	3.67	16	262
12:30	37987	37481	506	1.33	17.46	2.52	16	129
12:50	50538	49643	895	1.77	19.71	4.75	16	251
13:20	17706	17482	224	1.26	18.09	3.05	16	78
13:25	126338	123983	2355	1.86	17.72	2.64	16	123
13:30	65804	64824	980	1.49	17.39	2.21	16	76
13:35	18669	18407	262	1.40	17.66	3.30	16	157
13:45	134377	131477	2900	2.16	20.18	5.67	16	659
13:55	122372	120304	2068	1.69	17.95	3.13	16	136
14:20	59319	58639	680	1.15	17.26	1.75	16	82
14:20	6003	5868	135	2.25	18.78	3.98	16	104
14:20	46067	45485	582	1.26	17.70	2.40	16	82
14:20	54579	53246	1333	2.44	21.35	6.26	16	121
14:30	1707	1675	32	1.87	20.84	6.82	16	196
14:35	117399	115060	2339	1.99	18.28	4.33	16	669
14:40	29802	29379	423	1.42	18.41	3.42	16	115
15:15	91763	89501	2262	2.46	23.08	7.44	16	178
15:20	4633	4576	57	1.23	17.86	5.39	16	350
15:25	39795	39086	709	1.78	18.85	4.06	16	97
15:35	83466	82357	1109	1.33	17.95	3.05	16	80
15:55	47978	47394	584	1.22	18.92	4.61	16	193
16:06	48244	47314	930	1.93	20.27	5.76	16	147
16:10	120764	118574	2190	1.81	19.47	4.72	16	136
16:15	2993	2941	52	1.74	21.87	5.89	16	106
16:20	33220	32746	474	1.47	18.80	10.64	16	1807
16:20	6679	6573	106	1.59	20.76	4.74	16	173
16:30	46732	45265	1467	3.14	24.90	7.92	16	179
16:50	95347	94001	1346	1.41	18.21	2.91	16	85

An example of RTT on the path GIT-METU is shown in Figure 5.15.

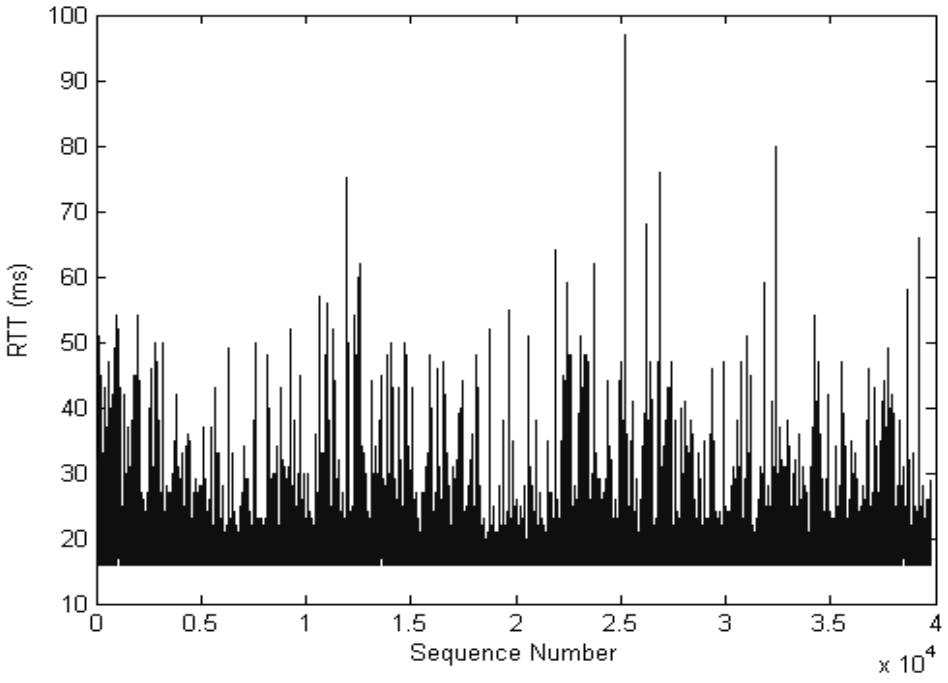

Figure 5.15 RTT on the path GIT-METU (03:25 PM)

Normalized histogram of the RTT shown in Figure 5.15 is plotted in Figure 5.16. Normalized histogram of the RTT and assumed gamma, lognormal and Weibull distributions are shown in Figure 5.17. Kolmogorov-Smirnov Goodness-of-fit test results for RTT measurements of the path GIT-METU are summarized in Table 5.9. We have observed that, for most of the RTT observations, assumed lognormal distributions returned less D values then assumed gamma or Weibull distributions. We concluded that RTT values on the path GIT-METU can be modeled with lognormal distribution with different parameters at different time of the day.

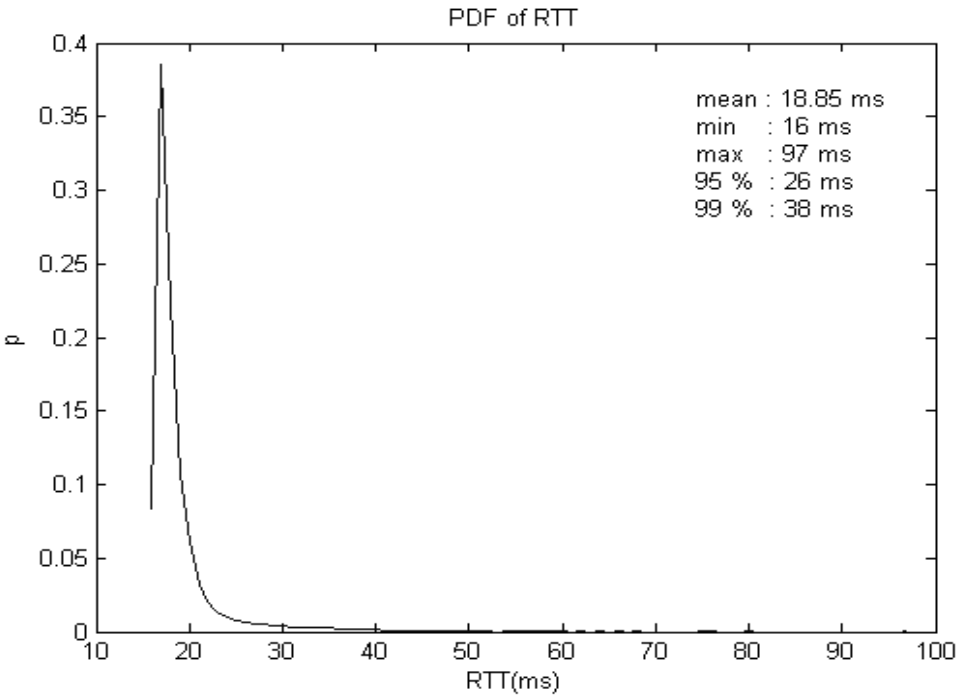

Figure 5.16 PDF of RTT on the path GIT-METU (03:25 PM)

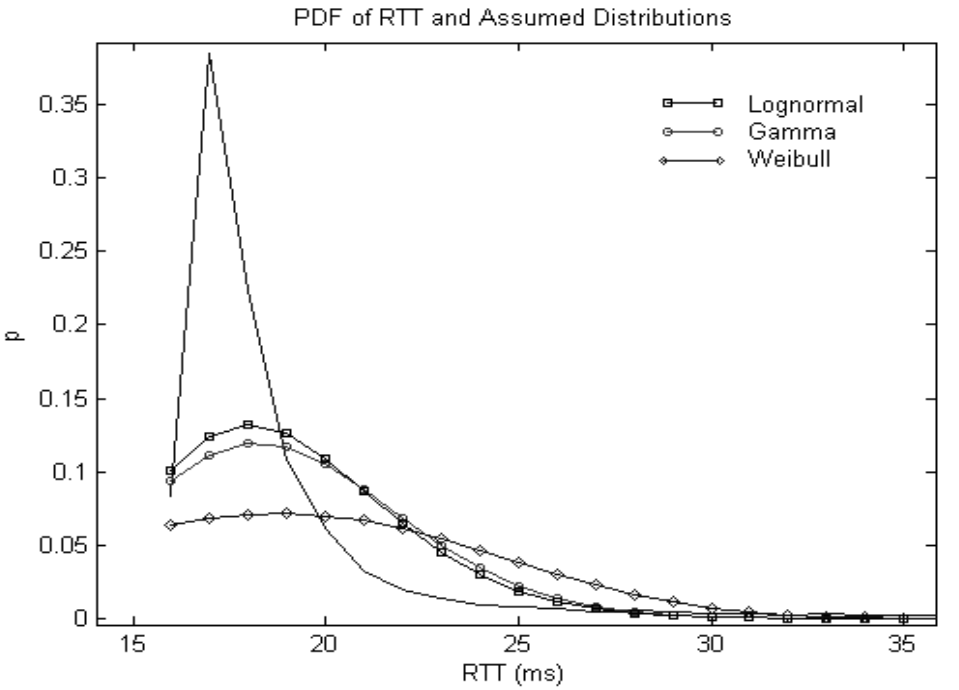

Figure 5.17 PDF of RTT and assumed distributions (detailed)

Table 5.9. Kolmogorov-Smirnov goodness-of-fit test results for RTT measurements on the path GIT-METU

Time of Day	Gamma		Lognormal		Weibull		D			D(α)
	α	β	μ	σ	α	β	Gamma	Logn	Weibull	
09:15	49.5679	0.3730	2.9071	0.1339	0	5.0000	0.2560	**0.2498**	0.2967	0.0052
09:25	40.8995	0.4534	2.9079	0.1454	0	4.4884	0.2621	**0.2554**	0.3150	0.0061
09:40	61.2170	0.2970	2.8922	0.1192	0	6.1840	0.2596	**0.2515**	0.2860	0.0056
10:35	30.8087	0.6197	2.9329	0.1682	0	3.9763	0.2448	**0.2409**	0.3028	0.0064
10:40	43.4281	0.4252	2.9043	0.1409	0	4.5147	0.2736	**0.2666**	0.3192	0.0046
10:45	36.2925	0.5196	2.9232	0.1555	0	4.2161	0.2496	**0.2448**	0.3045	0.0035
11:00	21.8022	0.9536	3.0114	0.2018	0	3.5946	0.2057	**0.1936**	0.2711	0.0057
11:10	39.8592	0.4678	2.9131	0.1462	0	4.3849	0.2648	**0.2596**	0.3120	0.0061
11:15	22.8595	0.8435	2.9371	0.1924	0	3.5793	0.2796	**0.2749**	0.3122	0.0058
11:35	55.4410	0.3271	2.8887	0.1253	0	5.6875	0.2641	**0.2548**	0.3046	0.0096
12:15	112.6363	0.1557	2.8600	0.0876	0	9.6770	0.2900	0.2929	**0.2691**	0.0080
12:20	72.0646	0.2467	2.8710	0.1072	0	6.8550	**0.2673**	0.2729	0.3048	0.0117
12:20	38.6101	0.4879	2.9228	0.1503	0	4.4573	0.2472	**0.2431**	0.2983	0.0126
12:25	9.5967	2.9061	3.2752	0.3288	0	3.4135	0.1380	0.1379	**0.1253**	0.0163
12:30	48.4535	0.3729	2.8839	0.1270	0	6.9074	0.2844	**0.2724**	0.2860	0.0041
12:30	91.3386	0.1911	2.8543	0.0928	0	11.527	**0.3013**	0.3049	0.2819	0.0070
12:50	25.0258	0.7876	2.9610	0.1865	0	3.8565	0.2415	**0.2310**	0.2771	0.0061
13:20	52.1032	0.3473	2.8860	0.1275	0	5.3615	0.2801	**0.2698**	0.3224	0.0103
13:25	78.1134	0.2268	2.8681	0.1023	0	8.4669	**0.2617**	0.2662	0.2721	0.0039
13:30	97.6103	0.1781	2.8506	0.0922	0	10.5284	0.3162	0.3165	**0.2867**	0.0053
13:35	69.1368	0.2555	2.8641	0.1030	0	9.8748	0.2755	0.2677	**0.2638**	0.0100
13:45	20.5676	0.9812	2.9802	0.2042	0	3.5794	0.2279	**0.2194**	0.2732	0.0037
13:55	59.5303	0.3015	2.8792	0.1168	0	6.9527	0.2673	**0.2549**	0.2874	0.0039
14:20	146.6242	0.1176	2.8439	0.0763	0	12.5029	0.3295	0.3285	**0.3126**	0.0056
14:20	35.0024	0.5365	2.9183	0.1545	0	4.2931	0.2583	**0.2540**	0.3110	0.0177
14:20	78.6366	0.2251	2.8671	0.1044	0	8.2633	**0.2772**	0.2802	0.2825	0.0064
14:20	15.5600	1.3722	3.0286	0.2395	0	3.2973	0.2202	**0.2085**	0.2523	0.0059
14:30	18.7632	1.1109	3.0102	0.2084	0	3.7833	0.1796	**0.1585**	0.2535	0.0331
14:35	46.1230	0.3964	2.8952	0.1292	0	6.0564	0.2741	**0.2662**	0.3018	0.0040
14:40	44.0641	0.4177	2.9013	0.1386	0	4.9678	0.2615	**0.2542**	0.3098	0.0079
15:15	13.3185	1.7333	3.1011	0.2613	0	3.4269	0.1615	**0.1559**	0.2006	0.0045
15:20	54.4512	0.3280	2.8733	0.1099	0	8.3980	0.2732	**0.2526**	0.2675	0.0200
15:25	31.4523	0.5994	2.9207	0.1644	0	3.8242	0.2703	**0.2650**	0.3240	0.0069
15:35	52.5361	0.3417	2.8781	0.1263	0	5.6928	0.2911	**0.2786**	0.3261	0.0047
15:55	30.8765	0.6129	2.9241	0.1626	0	4.2207	0.2578	**0.2547**	0.3040	0.0062
16:06	18.6375	1.0875	2.9820	0.2146	0	3.4843	0.2363	**0.2272**	0.2736	0.0062
16:10	26.0800	0.7465	2.9495	0.1805	0	3.8707	0.2495	**0.2374**	0.2942	0.0039
16:15	19.1216	1.1439	3.0589	0.2159	0	3.6119	0.1525	**0.1337**	0.2329	0.0250
16:20	32.3160	0.5818	2.9183	0.1488	0	4.5089	0.2438	**0.2412**	0.3031	0.0075
16:20	24.6097	0.8434	3.0124	0.1953	0	4.4976	**0.1764**	0.1772	0.1932	0.0167
16:30	11.6984	2.1287	3.1716	0.2859	0	3.3179	0.1431	**0.1354**	0.1656	0.0064
16:50	54.0370	0.3370	2.8928	0.1270	0	5.0127	0.2729	**0.2644**	0.3183	0.0044

We have observed that 39 out of 42 RTT observations fall into Class A distributions introduced by [17]. An example of normalized distribution of RTT falls to Class A is shown in Figure 5.18.

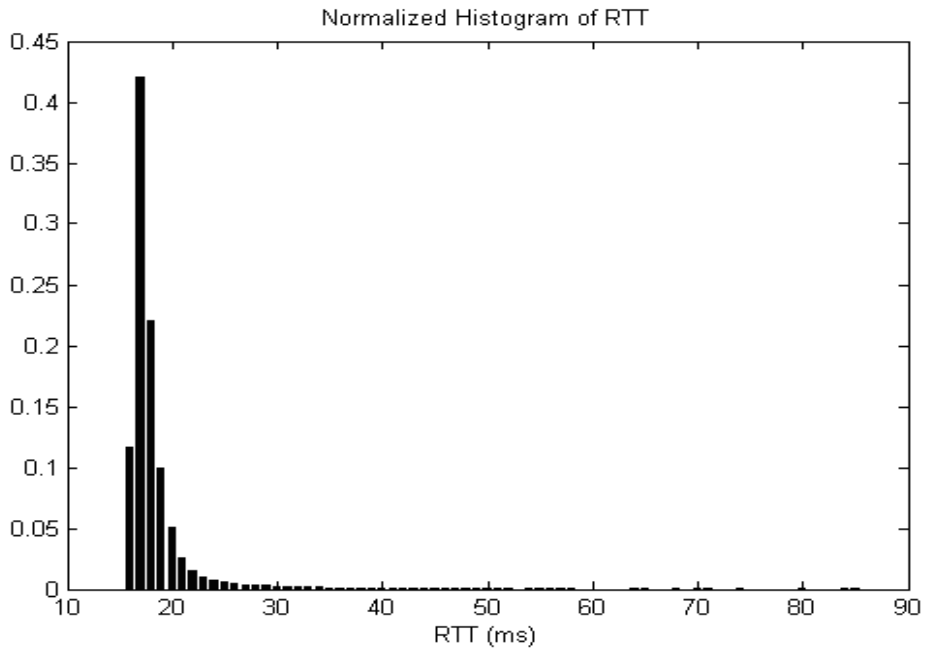

Figure 5.18 Example of normalized histogram of observed RTT of class A

We have observed that 1 out of 42 RTT observations fall into Class B distributions introduced by [17]. Normalized distribution of observed RTT that falls to Class B is shown in Figure 5.19. We have observed that 2 out of 42 RTT observations do not fall into any class introduced by [17]. These two normalized distributions are shown in Figure 5.20 and Figure 5.21.

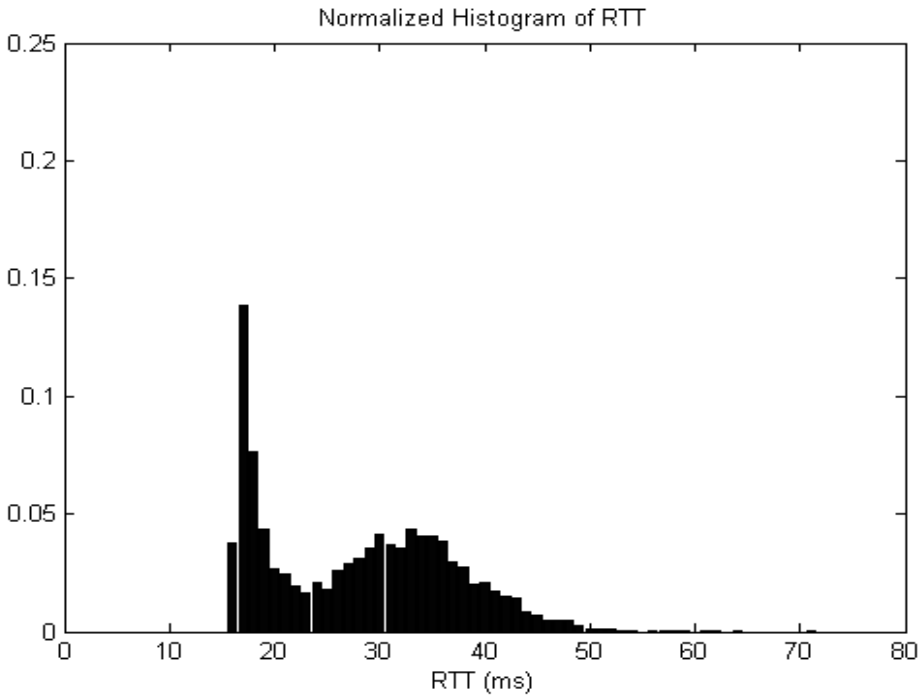

Figure 5.19 Normalized histogram of observed RTT of class B

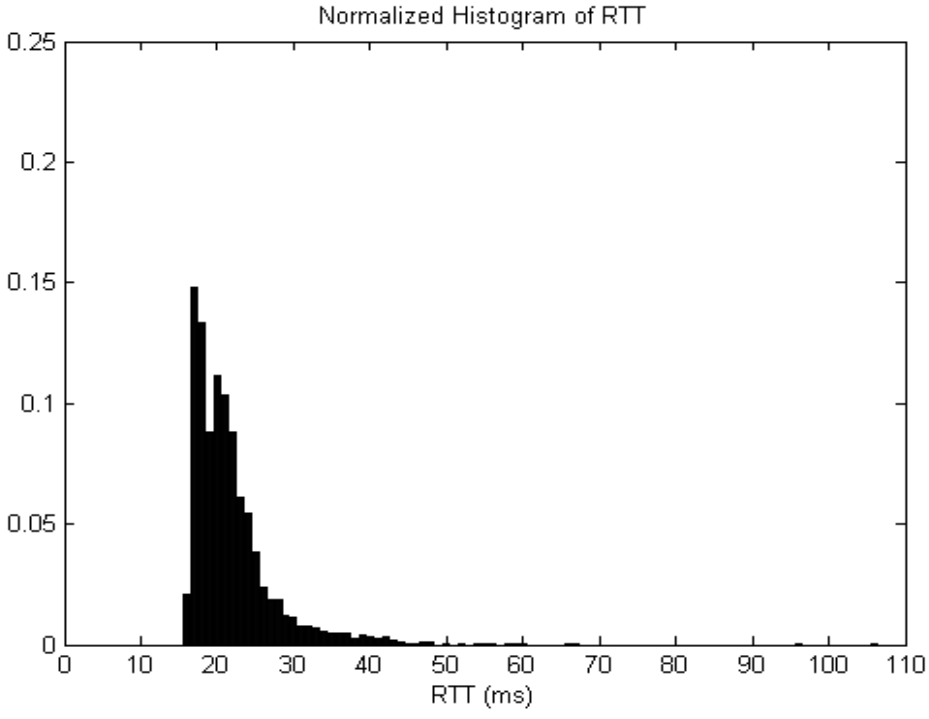

Figure 5.20 Normalized histogram of observed RTT (Uncategorized)

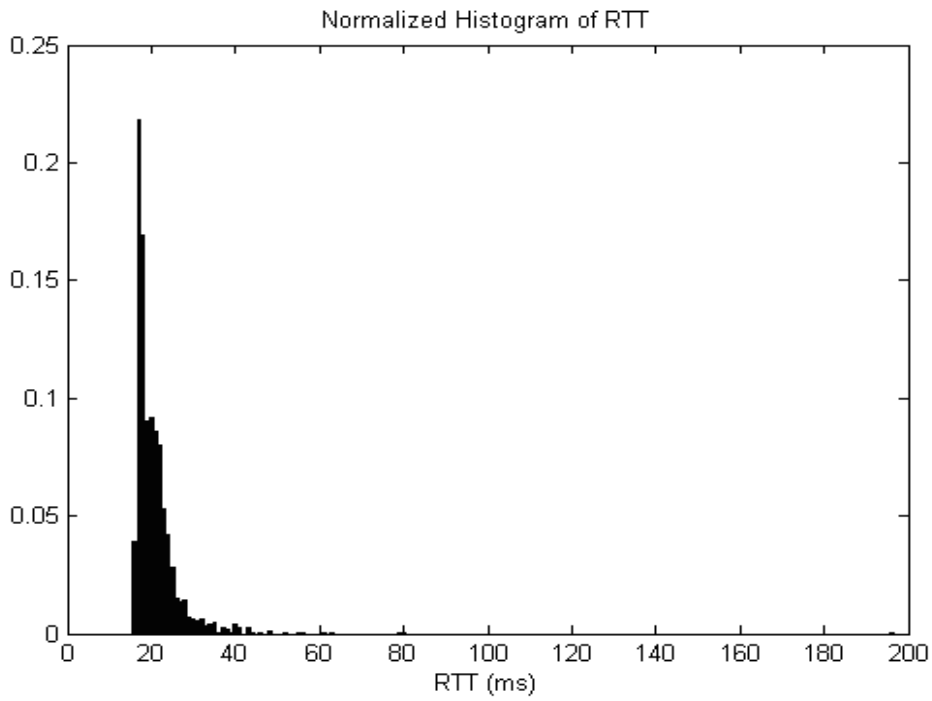

Figure 5.21 Normalized histogram of observed RTT (Uncategorized)

5.6 Packet Loss Analysis

Traditional Internet applications such as *telnet* and *ftp* use TCP as their transport layer protocol, and depend on TCP congestion control when there is congestion in the network. In TCP, a sender increases its transmission rate additively until it experiences a packet loss, which is taken as an indication of congestion. The sender then decreases its transmission rate multiplicatively, thus reacting quickly to the inferred congestion. As a result of this behavior, the transmission rate of the TCP sender is determined by the level of congestion in the network.

Continuous media applications such as VoIP, on the other hand, use UDP as their transport layer protocol and usually react to network congestion with less flexibility, due to their more stringent timing constraints. Continuous media applications can be loss-adaptive and delay-adaptive. Such adaptive applications keep track of packet

delays, and reflect any change in packet delays in their calculation of "play-out time." So, it is clear that packet loss and delay have tremendous impact on continuous media applications. Both loss and delay result from buffering within the network. As packets traverse the network, they are queued in buffers (thus adding to their end-end delay) and from time to time are dropped due to buffer overflow.

Consider a continuous media packet stream at a buffer that is filling up fast with packets from other traffic sources as well. The packets from the continuous media application continue to be queued up in the growing packet queue together with the packets from other sources. Continuous media packets arriving at the receiver experience progressively higher end-to-end delays than earlier packets. When the buffer reaches its capacity, packet losses begin to occur. The receiver of the continuous media application thus sees increased delay, and eventually losses. Figure 5.22 shows an example case for this scenario observed at our measurements.

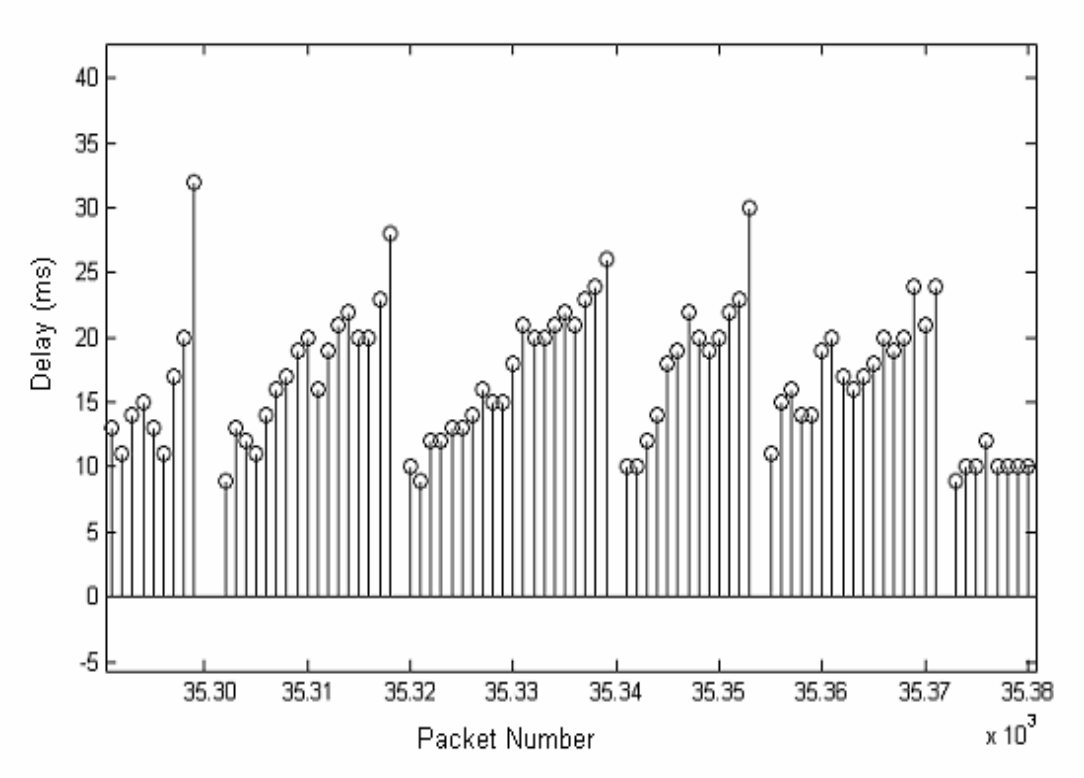

Figure 5.22 Packet losses after increasing delay

As we mentioned, we have taken one-way network delay and RTT measurements in this thesis. We have used packet numbers in order to identify the delays experienced by each packet. We can also determine the packet losses using these packet numbers. Losses in this work is only the *network losses*, losses caused by late arrivals in play-out buffering applications are not included.

It is observed that packet loss rate at one-way delay measurements changes from 4 % to 11 % and at RTT measurements from 11 % to 17 % on the path METU to HU (See Table 5.2 and 5.6). For the path GIT-METU, packet loss rate at one-way delay measurements is below 1 % and at RTT measurements below 2 % (See Table 5.6 and 5.8). As we have mentioned before, this dramatic difference at packet loss rates of the paths METU-HU and GIT-METU is caused because of the packet processing at the firewall of METU. We observed that most of the packet losses are single packet losses in one-way network delay and RTT measurements. Normalized histogram of packet losses observed at one-way delay measurements are shown in Figure 5.23.

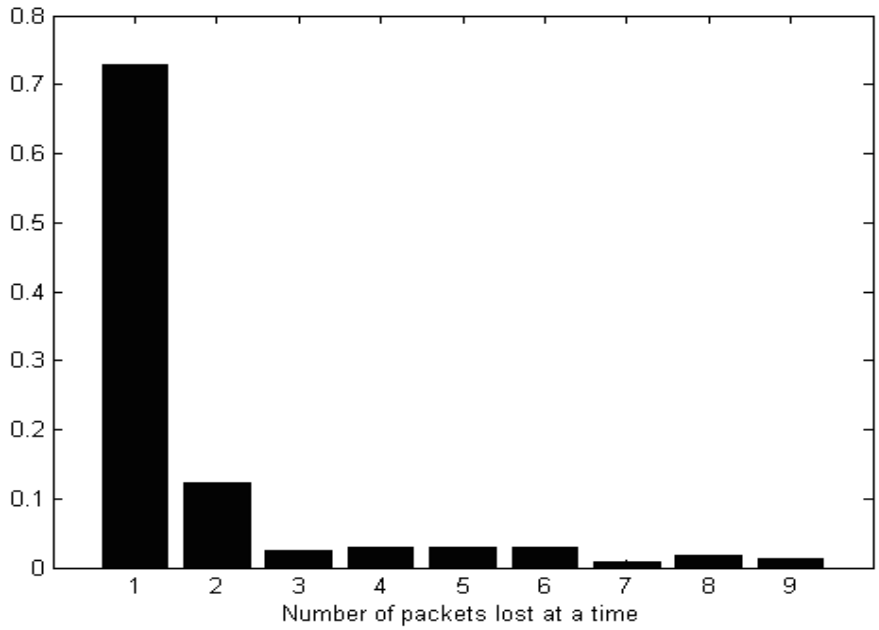

Figure 5.23 Normalized histogram of packet loss

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, we have attempted to determine the delay distribution in some segments of Turkish Internet network. We have measured Round Trip Time (RTT) and one-way network delay on two paths. The first set of measurements was on the path Middle East Technical University (METU)-Hacettepe University (HU) on 28-29 June 2003. The second set of measurements was on the path Gebze Institute of Technology (GIT)-Middle East Technical University (METU) from 27 October to 14 November 2003. In the first set of measurements, done on June 28-29, we have observed the negative effect of firewall used in METU. Before the second set of measurements was taken, the firewall rules were changed.

We have achieved 37 one-way delay measurements and 42 RTT measurements on the path GIT-METU. For one-way delay measurements, fixed size IP probe packets of 135 bytes are send with a period of 16 ms. For RTT measurements, fixed size IP probe packets of 128 bytes are send with a period of 50 ms. We have eliminated the effect of clock skew on one-way network delay measurements using Moon's linear programming algorithm by MATLAB. We have analyzed the distribution of RTT and one-way network delay using statistics toolbox of MATLAB.

We have observed that all of the distributions observed in one-way network delay measurements have a gamma-like shape with heavy tail which is classified as Class A distribution in [17]. For RTT measurements, 39 of 42 distributions have also the

same shape and fall into Class A, one distribution has a gamma-like shape with a gaussian lobe which is classified as Class B distribution in [17] and two distributions have a different unclassified shape. As a result, we have obtained familiar distributions as described in Test Traffic Measurements (TTM) of Reseaux IP Eurepeen Network Coordination Center (RIPE NCC).

We attempted to find out if observed distributions fit to gamma, lognormal or Weibull distributions. In order to do this, we have computed the maximum likelihood estimates of the parameters of these distributions for observations and applied Kolmogorov-Smirnov goodness-of-fit test to our observations and estimated distributions. None of the estimated distributions could pass the Kolmogorov-Smirnov goodness-of-fit test. But we have compared the closeness of the estimated distributions and observed distributions by the results of the Kolmogorov-Smirnov goodness-of-fit test.

We observed that, for the path METU-HU, distributions of one-way network delay and RTT can be estimated with Weibull distribution better than other distributions with different parameters at different time of the day. And for the path GIT-METU, distributions of one-way network delay and RTT can be estimated with lognormal distribution better than other distributions with different parameters at different time of the day. Difference in distributions observed on two paths is because of the firewall of METU.

We have measured packet loss during one-way network delay and RTT measurements. Packet loss rate on the path METU-HU was in the range 4 % to 11 % for one-way network delay measurements. Such high loss rate was because of the packet processing at the firewall of METU. Packet loss rate on the path GIT-METU was less than 1 %. We also observed that most of the packet losses were single packet losses. This observation agrees with Sanghi [12], who stated that losses generally occur one at a time.

We concluded that, the path GIT-METU satisfies the one-way network delay and packet loss rate requirements of VoIP.

The network interface of our measurement device is Ethernet and this was the main limitation in the measurements. We had to connect the measurement device to a LAN which is connected to the Internet via a gateway. We could attach the measurement tool to the LANs of Universities. Universities of Turkey are connected to the Internet via a dedicated network called ULAKNET. As we have connected the measurement device to METU, HU and GIT, our measurements were all on the ULAKNET. With this device we do not have any opportunity to measure delay on dial-up connections.

As a future work, the network interface of the measurement device can be adapted for dial-up networking, and delays on dial-up connections can be measured. Also the test-box of RIPE NCC can be used in order to obtain more accurate one-way network delay measurements.

REFERENCES

[1] N. S. Jayant, “Effects of packet losses on waveform-coded speech,” in Proceedings of the 5th International Conference on Computer Communications, (Atlanta, Georgia), pp. 275–280, IEEE, Oct. 1980.

[2] International Telecommunication Union (ITU), “Transmission systems and media, general recommendation on the transmission quality for an entire international telephone connection; one-way transmission time”, Recommendation G.114, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, Mar. 1993.

[3] R. Ramjee, J. Kurose, D. Towsley and H. Schulzrinne. “Adaptive play-out mechanisms for packetized audio applications in wide-area Networks”. In Proceeding of INFOCOM ’94 (1994).

[4] S. B. Moon, J. Kurose, and D. Towsley, “Packet audio play-out delay adjustment: Performance bounds and algorithms”. ACM/Springer Multimedia Systems 5 (January 1998), 17–28.

[5] S. B. Moon, P. Skelly, and D. Towsley, “Estimation and removal of clock skew from network delay measurement,” in Proceedings of IEEE INFOCOM ’99, pp. 227–234, March 1999.

[6] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: A transport protocol for real-time applications,” RFC 1889, January 1996.

[7] S. Savage, A. Collins, and E. Hoffman, “The end-to-end effects of Internet path selection,” in Proceedings of ACM SIGCOMM ’99, pp. 289–299, September 1999.

[8] V. Paxson, “End-to-end internet packet dynamics” in SIGCOMM Symposium on Communications Architectures and Protocols, (Cannes, France), Sept. 1997.

[9] V. Paxson, “Measurements and Analysis of End-to-End Internet Dynamics”, PhD Thesis, University of California at Berkeley, Berkeley, California, May 1997.

[10] A. Mukherjee, “On the dynamics and significance of low frequency components of Internet load,” Internetworking: Research and Experience, Vol. 5, pp. 163–205, Dec. 1994.

[11] J.-C. Bolot, “End-to-end packet delay and loss behaviour in the Internet,” in SIGCOMM Symposium on Communications Architectures and Protocols (D. P. Sidhu, ed.), (San Francisco, California), pp. 289–298, ACM, Sept. 1993. Also in Computer Communication Review 23 (4), Oct. 1992.

[12] D. Sanghi, A. K. Agrawala, O. Gudmundson, and B. N. Jain, “Experimental assessment of end-to-end behaviour on Internet,” in Proceedings of the Conference on Computer Communications (IEEE Infocom), (San Francisco, California), pp. 867–874 (7d.2), March/April 1993.

[13] N. F. Maxemchuk and S. Lo, “Measurement and interpretation of voice traffic on the Internet”, in Conference Record of the International Conference on Communications (ICC), (Montreal, Canada), June 1997.

[14] B. Dempsey, M. Lucas, and A. Weaver, “An Empirical Study of Packet Voice Distribution over a Campus-Wide Network”, 1994.

[15] www.ripe.net “Official web site of RIPE”.

[16] F. Georgatos, F. Gruber, D.Karrenberg, M. Santcroos, A. Susanj, H. Uijterwaal, R. Wilhelm, “Providing Active Measurements as a Regular Service for ISP’s” in Proceedings of the Passive and Active Measurements Workshop (PAM2001), Amsterdam, April 2001.

[17] C. J. Bovy, H. T. Mertodimedjo, G. Hooghiemstra, H. Uijterwaal, P. Van Mieghem, “Analysis of End-to-end Delay Measurements in Internet” in Proceedings of the Passive and Active Measurements Workshop (PAM2002), Ft.Collins, March 2002.

[18] T. Çelikadam, “Thesis: Design and Development of an Internet Telephony Test Device”.

[19] J. Postel, “User Datagram Protocol”, RFC 768, 1980.

[20] J. Postel, “Transmission Control Protocol”, RFC 793, 1981.

[21] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP:A Transport Protocol for Real-time Applications.” RFC 1889, Internet Engineering Task Force, January 1996.

[22] J.-C. Bolot, “Characterizing End-to-End Packet Delay and Loss in the Internet”.

[23] D. Mills, “Network time protocol(version 3): Specification, implementation and analysis,” Tech. Rep., Network Information Center, SRI International, Menlo Park, CA, March 1992.

[24] D.Sanghi, O.Gudmundsson, A.K.Agrawala, “Study of Network Dynamics”, Proceedings of 4th Joint European Networking Conference, Trondheim, Norway, pp.124-131, May 1993.

APPENDIX A

MATLAB CODES

In this thesis two MATLAB functions are generated for data analysis. The function named "dsinorder" implements Moon's skew elimination algorithm for one-way network delay measurements. The function named "compdata" extracts the normalized histogram of observed delay or RTT and computes the maximum likelihood estimates of the parameters of gamma, lognormal and Weibull distributions for this delay or RTT. Kolmogorov-Smirnov goodness-of-fit test is applied to extracted normalized histogram and assumed distributions. Functions from statistics toolbox of MATLAB are used in these user defined functions. Both functions are given below.

```
function [dsresult] = dsinorder(dl)
%
% This function implements Moon's Algorithm for skew removal.
%
sequence = dl(:,1);
dly = dl(:,2);
length1 = length(dly);
X = [1:length1]';
X = X-1;
min_dly=min(dly);
max_dly=max(dly);
count= max_dly-min_dly+1;
figure, plot(sequence, dly), xlabel('Sequence Number'), ylabel('Network delay
(ms)'), title('ONEWAY NETWORK DELAY WITH CLOCK SKEW');
dc = min(dly) + 0.5;
diff_min = sum(dly);
slope = 0;
dc_shift = 0;
```

```
SS = 0;
while SS < dc,
   alfa = 0;
   while alfa < 0.001,
      Z = (X* alfa) + SS;
      diff1 = dly - Z;
      if min(diff1) >= 0
         if sum(diff1) < diff_min
            slope = alfa;
            dc_shift = SS;
            diff_min = sum(diff1);
         end
      else
         alfa = 0.001;
      end
      alfa = alfa + 0.00001;
   end
   SS = SS + 0.01;
end
line = slope * X + dc_shift;
hold on, plot(sequence,line, 'r'), text(12500,7,'Estimated Skew Line');
hold off;
dly = dly - (X * slope);
dly = ceil(dly);
figure, plot(sequence,dly);, xlabel('Sequence Number'), ylabel('Network delay
(ms)'), title('ONEWAY NETWORK DELAY WITHOUT CLOCK SKEW');
dsresult = [sequence dly];
```

function [hist_dly] = compdata(dl)

```
%
%In this function, normalized histogram of observed delay or RTT is extracted.
%Maximum likelihood estimates of the  parameters of gamma, lognormal and
%Weibull distributions for this delay or RTT are computed. Kolmogorov-Smirnov
%goodness-of-fit test is applied to extracted normalized histogram and assumed
%distributions.
%
dly = dl(:,2);
sequence = dl(:,1);
length1 = length(dly);
last = sequence(length1);
lost = last - length1;
loss = 100*(lost/last);
mean_dly = mean(dly);
std_dly = std(dly);
min_dly=min(dly);
max_dly=max(dly);
%
```

```
%%%%%%%%%%%%   PDF calculation of the input data %%%%%%%%%%
count= max_dly-min_dly+1;
hist_dly = hist(dly,count)/length1;
T = [min_dly:1:max_dly]';
%
%%%% Parameter estimation of GAMMA, LOGNORMAL and WEIBUL %%%%
gampar = gamfit(dly);
gamm_fit_dly=gampdf(T,gampar(1),gampar(2));
logpar1 = mean(log(dly));
logpar2 = std(log(dly));
log_fit_dly=lognpdf(T,logpar1,logpar2);
weibpar = weibfit(dly);
weib_fit_dly=weibpdf(T,weibpar(1),weibpar(2));
%
%%%%%%%%%%%%%%%%%% PLOT of PDFs %%%%%%%%%%%%%%%%
figure, plot(T, hist_dly), xlabel('Network delay (ms)'), ylabel('Probability'),
title('PDF OF DELAY WITHOUT CLOCK SKEW');
hold on;
plot(T,gamm_fit_dly,'r--');
plot(T,log_fit_dly,'k-.');
plot(T,weib_fit_dly,'b:');
hold off;
%
%%% CDF calculation of input data, GAMMA, LOGNORMAL and WEIBUL %%
[yCDF,xCDF,n,emsg] = cdfcalc(dly);  %%CDF calculation of input data
GCDF = gamcdf(xCDF, gampar(1),gampar(2));
LCDF = logncdf(xCDF,logpar1,logpar2);
WCDF = weibcdf(xCDF,weibpar(1),weibpar(2));
length2=length(yCDF);
yCDF = yCDF(2:length2);
%
%%%%%%%%% KOLMOGROV's GOODNESS of FIT test %%%%%%%%%
[Hg,Pg,KSSTATg,CVg] = KSTEST(dly, [xCDF GCDF]);
[Hl,Pl,KSSTATl,CVl] = KSTEST(dly, [xCDF LCDF]);
[Hw,Pw,KSSTATw,CVw] = KSTEST(dly, [xCDF WCDF]);
H = [Hg Hl Hw Hp];
D = [KSSTATg KSSTATl KSSTATw KSSTATp];
%
%%%%%%%%% OUTPUT FILE %%%%%%%%%%%%%%%%%%%%%%
fid=fopen('result.txt','w');
outfile1 = [last length1 lost loss mean_dly std_dly min_dly max_dly gampar(1)
gampar(2) logpar1 logpar2 weibpar(1) weibpar(2) KSSTATg KSSTATl KSSTATw
CVg];
fprintf(fid,'%3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t
%3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t %3.4f\t\n',outfile1);
fclose(fid);
```