KUNGL
TEKNISKA
HÖGSKOLAN

Wide Area
Measurements of Voice over IP Quality

Master Thesis in Computer Science

by Fengyi Li

Supervisor: Ian Marsh
Examinator: Professor Gunnar Karlsson

Department of
Microelectronics and Information Technology (IMIT)
Royal Institute of Technology(KTH)
Stockholm, Sweden
June 2002

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Ian Marsh, for the guidance and inspiration he provided me during this work. I have not only gained knowledge from him, but also methods with which to solve problems, which I will benefit from in the future.

Special thanks to professor Gunnar Karlsson for his valuable suggestions.

Many thanks go to all the people at TTS/LCN lab for their help and support: Marcello Conte, Evgueni Ossipov, Nacho “Más” Ivars, Hector Velayos, Henrik Lundquist, Gyorgy Dan, Lena Wosinska, Viktória Fodor, professor Rolf Stadler and George Koudouridis. I would also like to thank the system administrator Enrico Pelleta for his kind support.

Many thanks also go to our remote colleagues who provided us access to their computing resources plus for assisting us by answering our queries.

Abstract

Voice over IP (VoIP) is an attractive alternative to the traditional telephone system. However, the Internet was not designed to carry real-time data. The transmission of real-time data including voice, may experience problems due to the packet switched nature of the Internet and competing traffic. In this work, we have measured three quality parameters, packet loss, delay and jitter for ten hosts around the world to investigate VoIP quality issues. Our goal is to assess the suitability of VoIP in 2002 and have gathered nearly 25,000 sample sessions towards this goal. This is a second attempt to measure VoIP quality, the last one was in 1999. We conclude that, as in 1999, VoIP is still feasible although still not on a global scale. We also give results on the use of silence suppression, routing asymmetry and the effect of packet size on the measurements.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Voice over IP (VoIP) is the transmission of voice signals as packets of data using the Internet Protocol. The voice information is sent in digital form in discrete packets rather than by the traditional circuit-committed architecture of the public switched telephone network (PSTN). The biggest advantage of IP telephony is the elimination of long-distance telephone charges [1], and therefore IP telephony is rapidly gaining acceptance [2]. However, quality issues remain a big problem.

Voice over IP requires real-time handling. Using the PSTN, voice travels over a circuit-switched network, which establishes a dedicated channel that remains connected for the duration of the call, therefore the PSTN can allocate resources for acceptable quality. In contrast, IP networks, using a best effort service, cannot guarantee the performance needed for real-time transmission and reception. IP-based networks use packet switching that routes packets over many paths. Data can be lost, delayed, and must compete with web transfers and music downloads. It is well known that VoIP users are susceptible and sensitive to quality changes, causing them to abandon calls when the quality deteriorates below an acceptable level [3].

Our goal is to investigate the quality issues of VoIP to see the suitability of IP telephony on today's Internet. Therefore, we have performed a series of wide-area experiments. The particular quality characteristics we look at are packet loss, delay and jitter. These empirical parameters are used to ascertain the quality of the network for VoIP transmissions. They are also useful input to develop reliability mechanisms such as error correction and concealment. Moreover, they can help administrators to tune the network for optimal VoIP performance if so desired.

This thesis is organized as follows: Chapter 2 describes the measurement infra-structure, including the measurement tool, the co-operating sites, how the trace files are post-processed plus communication details between the hosts. Chapter 3 presents in detail the measurement results, including loss, jitter and delay. We also discuss the goals of measuring the individual parameters and some considerations of the network, including the effect of silence suppression, network asymmetry and other tools for measuring the network condition. Chapter 4 lists some related work and compares our results with those presented. Chapter 5 describes the work that could be done if this work is to be continued. Chapter 6 is a summary of what we have learned by doing this work. Finally, Chapter 7 summarizes this work.

Chapter 2

Simulating Real Calls

In this chapter, we describe how simulated telephone calls are used to measure the quality of intervening links. We also describe the measurement infra-structure, which consists of ten co-operating sites with differing properties between each other. By properties we mean geographical distance, number of hops, round trip time, link bandwidths etc. We use an audio tool, Sics*o*phone [4], to transmit packets between each of the co-operating sites in order to gain the measurements.

2.1 The Measurement Tool: Sics*o*phone

We use Sics*o*phone as the tool for our measurements. Sicsophone is a VoIP application, developed by Ian Marsh and Olof Hagsand at SICS, and Kjell Hanson at Ericsson Business Networks. It delivers audio to two communicating parties using the RTP and RTCP [5] protocols. The main focus of the tool was low-delay audio communication, but it has been substantially modified for our measurement purposes. There are two main reasons why we chose to use pre-recorded calls instead of real calls: one is that real calls at the remote sites is not always possible, people are not always available to talk with and the second is that we need to do the test at regular intervals, this is, of course, much easier using an automated procedure. The drawbacks are it not possible to say exactly how the quality is perceived and the call duration is fixed rather than varying as with “natural” speakers.

2.2 Co-operating Sites

In addition to our local host in Stockholm, we have ten co-operating sites around the world. The central point does not need to be in Sweden, it is simply because we are located here. The choice of a central location can however be important, in that the choice of timestamping the data files can be done either at a centralized site or at the ’local’ receiver. We have made the infra-structure so that the files are stamped according to our central node, this is discussed more in Section 2.6.

Our hosts are all computers at universities. Table 2.1 lists their names, locations and operating systems.

Host Name	Location	OS
veloso.cs.umass.edu	Amherst, Massachusetts, US	Linux 2.4.16
idmaps.eecs.umich.edu	Ann Arbor, Michigan, US	FreeBSD 4.3
rain.cs.ucla.edu	Los Angeles, US	FreeBSD 2.5
obo02.dmz.fundp.ac.be	Namur, Belgium	Linux 2.4.2
topor.atm.tut.fi	Tampere, Finland	Linux 2.4.17
login.sm.luth.se	Lulea, Sweden	SunOS 5.8
cost.fokus.gmd.de	Berlin, Germany	SunOS 5.8
mikrodalga.org	Istanbul, Turkey	Linux 2.4.7
zorzal.dc.uba.ar	Buenos Aires, Argentina	SunOS 5.7
yoyo.cs.waikato.ac.nz	Hamilton, New Zealand	Linux 2.2.16

Table 2.1: Sites and Hosts Available

The geographical locations of the hosts for the measurements are shown in Figure 2.1.

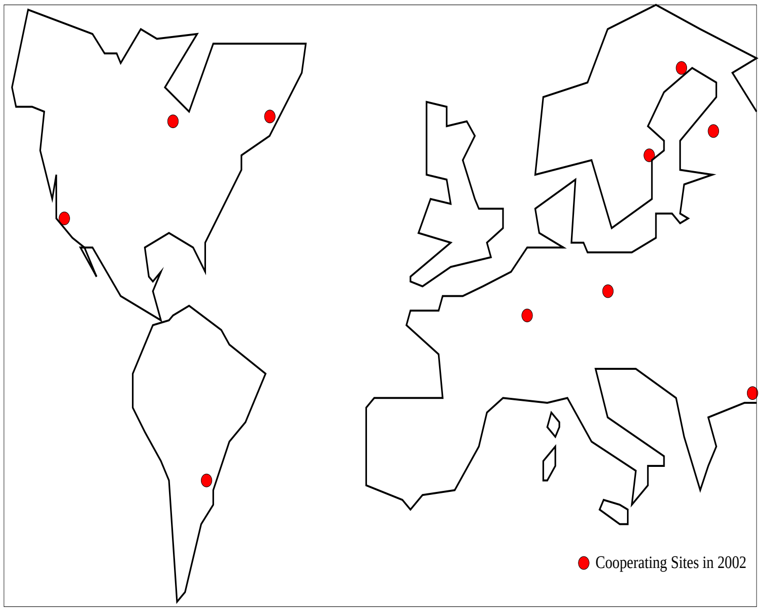


Figure 2.1: Locations of Test Sites (New Zealand host only added recently)

We have chosen these hosts due to the big variations in number of hops and geographical distances. Table 2.2 shows the number of hops and time difference between each of the hosts. Within this report, tables of this form are sender down the rows and receiver across the columns.

The top number in the cell is the number of hops from the sender to the receiver. It is obtained by using traceroute over a period of time, hence on a number of occasions. There is a variation in the number of hops reported by traceroute, exactly the reason why, we cannot say, but one possibility is changes in router configurations. Due to root access restrictions (some hosts

receiver sender	**Mass.**	**Mich.**	**Cali.**	**Belg.**	**Finl.**	**Swed.**	**Germ.**	**Turk.**	**Arg.**	**NZ**
Mass.	/	14+1 0	19 -3	11 +6	15 +7	21 +6	17+3 +6	20 +7	20 +1	18 +16
Mich.	14+1 0	/	20+1 -3	11 +6	17 +7	23 +6	16+1 +6	20 +7	25 +1	22 +16
Cali.	18+1 +3	21 +3	/	20 +9	25+1 +10	30+2 +9	23 +9	23 +10	25 +4	20 +19
Belg.	16 -6	17 -6	23 -9	/	17 +1	22 0	13 0	16+2 +1	19 -5	19 +10
Finl.	15+1 -7	17+1 -7	24+2 -10	16 -1	/	20+1 -1	17+2 -1	19 0	23 -6	22 +9
Swed.	22+1 -6	25 -6	30 -9	24 0	21 +1	/	25 0	26 +1	41 -5	30 +10
Germ.	15 -6	16 -6	22 -9	12 0	17 +1	22 0	/	16 +1	18 -5	16 +10
Turk.	18+1 -7	20 -7	19 -10	17 -1	19 0	25 -1	16 -1	/	18 -6	18+4 +9
Arg.	NA -1	NA -1	NA -4	NA +5	NA +6	NA +5	NA +5	NA +6	/	NA +15
NZ	16 -16	23 -16	20+2 -19	21+4 -10	29+1 -9	30 -10	20 -10	25 -9	24 -15	/

Table 2.2: Connectivity (hops and time differences)

require users to be root to access tools like traceroute) the number of hops are not always known, we mark them with a 'NA' (not available). The bottom number is the time difference. For instance, the difference between the time in Massachusetts (EDT) and the time in California (PDT) is 3 hours, the minus indicates California is behind.

2.3 Remote Access

We use secure shell, SSH, for remotely logging into the hosts and starting the pre-recorded call. SSH provides secure and encrypted communication between the hosts. Moreover, SSH allows logging in without passwords using public and private keys. This makes it possible to start measurements automatically and securely. More practical details on remote access with SSH are explained in Appendix A.

2.4 Measurement Infra-structure and Post-processing

2.4.1 Improvement to the Existing System

As stated Sicsophone was used for VoIP quality measurements in 1999 for the first time. We have made some improvements from the experiences we obtained three years ago.

- The most important improvement is better utilization of the links by using a full-mesh topology. The last measurements used a centralized topology with Stockholm as the co-ordinating point. This time we use a full-mesh topology, in which we fully inter-connect the hosts available to us. Ten hosts connected to each other allows us 90 interconnections, while centralized topology gives only nine inter-connections. Moreover, a full-mesh topology makes it possible for bidirectional tests, needed for detecting VoIP quality asymmetry.
- Automatic invocation. Sicsophone was started manually for setting up conversations. Now it runs automatically initiated by UNIX shell scripts, this makes it possible for continuous long-term tests.

2.4.2 Measurement Infra-structure in 2002

As said, the initiation of the session between each pair of the machines is done by a shell script which runs from the local machine. The measurement is done in the following steps:

1. Remotely login to the receiving machine and start the receiving process.
2. Remotely login to the sending machine and start the sending process. The sender is started after the receiver to synchronize the receiver.
3. Wait until the session finishes. Normally it takes about two and a half minutes using a packet size of 160 bytes.
4. Copy the log file from receiving machine to the central machine. We collate all the log files on the local machine for post-processing.

To perform the tests over a long period of time, we need to initiate the session at given times. This is done by *crontab*, which is used for scheduling jobs to let them run at particular times. In our tests, a session between a certain pair of hosts is scheduled once per hour. We chose this frequency as it does not place high loads on remote machines which we do not own. As stated all the machines can send and receive to/from each other.

The advantage of using scripts with Sicsophone is that other audio tools could be used instead of Sicsophone. For example a tool which supports a coding style which we cannot (real audio for example). The scripts simply act as a “wrapper” for any given audio tool.

2.5 Post-processing

We use both Perl and Java for post-processing the log files to obtain the required loss, delay and jitter values. Perl is used as it handles multiple files more effectively than Java so calculations such as the average loss across all recorded sessions are done using Perl. Operations on single or two files (e.g. the loss for an asymmetric route) or more complex mathematical operations have been programmed in Java.

2.6 Log Files

We have until now, gathered 24,770 log files and the measurements are still continuing. Therefore we need a suitable way to store and to process them. They are also intended to be used for further investigations both by us and other researchers. [1].

The files are gathered under a top level directory (called Trace) then each host-pair is represented as a directory. The hostname is used rather than the location as different hosts in the same location might be used. Table 2.1 provides the mapping from name to geographic location. Each session is stored under the host-pair name as a single, compressed text file. So for example, 'rain_obo02' contains the file '200203140130_1_20.log.gz' which is record of the session, from its inception to its completion. The filename is the date, whether we use silence suppression (1=yes) and the packet size in ms (20=160 bytes).

2.6.1 Log Files Format

One advantage of using Sicsophone is its ability to save detailed log files. For each transmission, Sicsophone creates a log file at the receiver side, recording each packet event, the silence periods are also recorded, as an 'empty' event. The trace file format used in this work is essentially a dump of the RTP header but Sicsophone can also give, playout buffer timings, talkspurt information, round-trip time estimates, RTCP reports and so on. An example of the log file format used in this work is shown in Figure 2.2.

```
I 1017112866 661181
T 1017112866 862086
E 1017112866 883304 172 32869 0 0 160
E 1017112866 902002 172 32869 1 0 320
E 1017112866 917461 172 32869 2 0 480
E 1017112866 936997 172 32869 3 0 640
--------------------------------------
1 2          3      4   5     6 7 8
```

Figure 2.2: Log File Format

'I' indicates a session start, 'T' a timeout, and 'E' data has arrived. The fields numbered 1-8 are explained in Table 2.3.

Field 5 is primarily used to inform the receiver about the type of media it is currently receiving. It is a combination of the information in RTP header.

[1] Available from http://www.sics.se/∼ianm/COST263/Trace

Field No.	Meaning
1	Event Type I, T, E
2	Epoch seconds
3	Epoch Microseconds
4	Packet Size (incl RTP hdr)
5	RTP hdr
6	Sequence number
7	Timestamp (incremented when 8 wraps)
8	Timestamp

Table 2.3: Log File Fields

The sequence numbers and timestamps are used for calculating the packet loss and jitter, they are explained in the next chapter. Moreover, with Sicsophone we can choose to send packets of different sizes and with or without silence suppression to see their affects on the quality. Sicsophone supports multiple sessions between a sender and receiver at the same time using different port combinations, so we can compare sessions with different properties by simultaneously sending them between the same host-pair (e.g. the silence suppression and packet size were recorded in this way).

2.6.2 Log File Timestamps

As stated, we save all the log files in subdirectories named by the sender plus receiver in the form of host1_host2. The file name is the timestamp at the central location (Sweden) when the sessions were started. Swedish time is GMT $+1$ hour for winter time and GMT $+2$ for summer time. This makes it easy to compare files taken at the same time, for example, when investigating asymmetry. We can just look for the files with the same name but in different directories, one named by host1_host2 and another named by host2_host1.

For plots when the local time is important (i.e. time of day) then we can use the time in the files themselves (the UNIX Epoch) to get the local time at a particular location, this was done for the plots where the time of day is important.

Chapter 3

Results

We have looked at the quality parameters for VoIP: loss, delay and jitter. In this chapter, we give definitions of these values, discuss the goals of the measurements and present the results of our 15 week tests. Additionally, we have investigated the effects of network asymmetries and silence suppression on the measurement results.

3.1 Packet Loss

3.1.1 Definition of Loss

We define loss as number of packets that were sent from the sender that did not arrive at the receiver. We are normally interested in the packet loss percentage, therefore we look at the percentage of missing packets from the total number of packets expected. We do not include late packets as lost, this is consistent with [5].

3.1.2 How We Calculate Loss

We use information in the session record (i.e. Sicsophone's trace file) to calculate the packet loss. The sequence number starts from zero and is incremented by one for each packet received. Packet loss is detected via the sequence number incrementing by more than one, thus by observing the sequence number we can calculate the total loss. The last sequence number plus one is the total number of packets sent, since the sequence number started from zero.

$loss\% = total\ number\ of\ packets\ lost/(last\ sequence\ number\ +\ 1)$

As a simple example, figure 3.1 is a portion of a Sicsophone log file. In this case, packets 3 and 5 were lost, so the loss percentage is $2/8 = 25\%$. We could have used RTCP to record the loss, but we wanted more granularity about the loss structure than the RTCP sender and receiver reports can give us. We were also interested in the duration of lost packets, this is not possible to obtain via intermittent "quality" reports (or even the goal of them).

```
T 1021131418 180020
E 1021131418 228925 172 32869 0 0 160
E 1021131418 250200 172 32869 1 0 320
E 1021131418 273070 172 32869 2 0 480
E 1021131418 313876 172 32869 4 0 800
E 1021131418 355274 172 32869 6 0 1120
E 1021131418 376094 172 32869 7 0 1280
```

Figure 3.1: Log File Example

3.1.3 Importance of Loss

- To estimate how the quality would be perceived by a human. Lost packets undeniably deteriorate the quality, but by how much and in which situations is not quite so well known. Also it can be codec dependent as different codecs have different tolerances for packet loss. For example, PCM is sensitive to packet loss, more than 1% may cause significant impairments [6], while GSM is much more tolerant to loss, and in fact was designed to be robust against lost data. In our tests, we use PCM coding, and often quote 10% as the upper limit as a significant loss in quality. Also schemes such as forward error correction (FEC) mechanisms [7] cease being useful with loss rates over 10%.

- Quality value. A better method to judge the overall quality is to use the ITU-T's E-model [8]. The E-model is a tool for estimating the voice quality (see section 4.1). It combines delay, loss, jitter, and coder frame size into a single value, the R-factor, which yields the level of the quality. Our measurements can give input to calculating this value.

- Rate adjustment. A sending rate which is too high may cause congestion in the network, thus causing loss. With knowledge of the packet loss, the sending rate can be monitored, hence during high loss periods the sending rate can be reduced.

- For monitoring purposes. Packet loss usually occurs when there is congestion on the packets' path, often 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 [9] mechanisms. The packet loss is important for knowing if congestion is occurring in the network and for developing better queuing algorithms in the routers.

- To use mechanisms such as Forward Error Correction (see Section 3.1.4) for protecting against adverse conditions.

3.1.4 Loss Results

Overall Loss Behavior

We record the loss percentage for each transmission. Figure 3.2 shows a typical loss pattern over one day for two different connections. The x-axis is the time of day from 0:00 to 24:00, and the y-axis is the loss percentage.

From the almost 25,000 measurements we have taken, the average loss rate is 0.84% with a standard deviation of 2.7%. 66.0% conncections showed zero

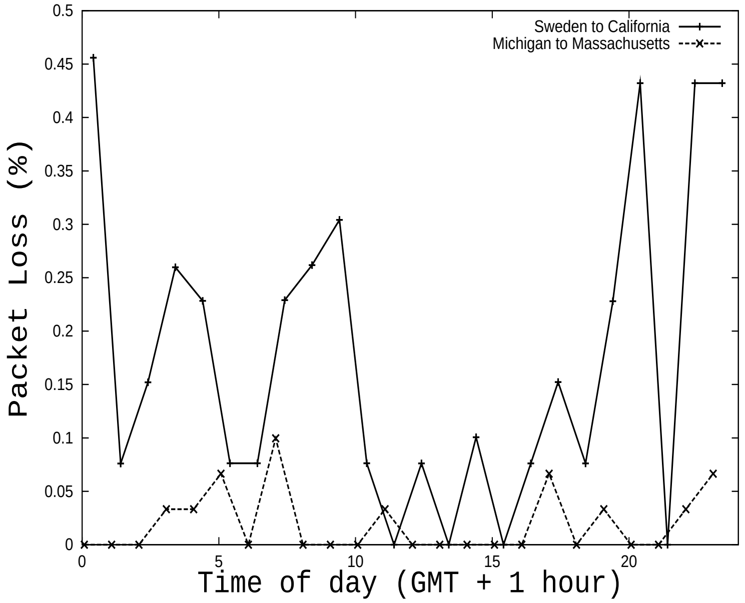


Figure 3.2: 24 Hour Loss

loss, 93.6% have a loss rate less than 5% and 98.2% have a loss rate less than 10%. This is of course dependent on the individual connection.

Table 3.1 shows the loss for connections between all the sites. The top number in the cell is the average loss percent and the lower one is the standard deviation. From this table we can see that most of the connections have very low loss rates (less than 1%) except the connections where the Turkish or Argentinian sites are involved. We can attribute the low loss to the high bandwidth networks which are rarely congested (Nordunet/Sunet) whilst the connection to Turkey experiences congestion due to either many users at the Turkish site, i.e. access problems or congested links. It should be noted from our measurements we cannot exactly say in which of the two cases where the loss occurs.

Time Effects

In earlier works [7, 10] it has been observed that the time of day and the day of week influence the packet loss. We have also observed that the loss rate is often time-dependent. Generally, during work hours and weekdays the observed loss was higher. However, because all our hosts are in universities, this conclusion is only for academic links. Due to the explosion of the use of the Internet in homes it is not clear whether this is also the case for commercial links. We speculate however the opposite is true but not have experimental evidence to support this claim.

- *Time of Day* During the busy hours of day packet loss is higher than during the night. Figure 3.3 depicts this.

receiver sender	Mass.	Mich.	Cali.	Belg.	Finl.	Swed.	Germ.	Turk.	Arg.
Mass.	/	0.066 —— (0.60)	0.121 —— (0.99)	0.124 —— (0.83)	0.096 —— (0.76)	0.037 —— (0.20)	0.0 —— (0.0)	4.88 —— (4.65)	8.971 —— (7.21)
Mich.	0.021 —— (0.15)	/	0.186 —— (1.10)	0.020 —— (0.11)	0.099 —— (1.09)	0.126 —— (2.19)	0.186 —— (0.90)	2.98 —— (1.92)	6.54 —— (7.06)
Cali.	0.057 —— (0.26)	0.122 —— (1.93)	/	0.196 —— (0.75)	0.604 —— (1.38)	0.191 —— (0.26)	2.822 —— (3.01)	4.367 —— (2.42)	8.932 —— (8.20)
Belg.	0.0 —— (0.0)	0.0 —— (0.0)	1.170 —— (0.97)	/	0.004 —— (0.01)	0.0 —— (0.0)	0.171 —— (0.66)	3.768 —— (2.67)	NA
Finl.	0.001 —— (0.10)	0.017 —— (0.25)	0.660 —— (1.39)	0.072 —— (0.25)	/	0.001 —— (0.01)	0.0 —— (0.0)	3.177 —— (1.73)	7.494 —— (6.49)
Swed.	0.0 —— (0.0)	0.025 —— (0.37)	0.069 —— (0.10)	0.081 —— (0.25)	0.001 —— 0.01	/	0.0 —— (0.0)	2.984 —— (0.95)	NA
Germ.	0.0 —— (0.0)	0.002 —— (0.01)	2.515 —— (1.87)	0.004 —— (0.01)	0.0 —— (0.0)	0.0 —— 0.0	/	3.732 —— 2.48	NA
Turk.	8.129 —— (2.77)	7.960 —— (2.91)	7.676 —— (6.83	7.093 —— (3.98)	7.786 —— (2.70)	8.389 —— (3.14)	7.965 —— (3.10)	/	NA
Arg.	0.506 —— (1.38)	0.521 —— (1.46)	0.618 —— (1.76)	0.480 —— (1.38)	0.529 —— (1.39)	0.028 —— (0.09)	0.056 —— (0.10)	5.822 —— (2.97)	/

Table 3.1: Packet Loss (Percentage and Deviation)

- *Day of Week* We have seen more loss during weekdays than weekend although the differences are generally not large. Examples are from Michigan to California the average loss during weekdays is 0.23%, while during weekends it is 0.13%. A second example from Turkey to Finland where the loss rate was 7.94% for weekdays and 7.61% for the weekends. Since the loss rate is still high for the connection to Turkey over the weekend we can say it is the *links* which are congested, as the university campus is probably not heavily used at the weekend.

Effect of Packet Size

We have investigated the effect of packet size on loss rates by running two simultaneous calls with different packet sizes. The packet sizes tested were between 160 and 1280 bytes. No obvious relationship between packet size and loss was observed. Figure 3.4 shows the differences of loss rate with size 160, 640 and 1280 bytes, sent from Massachusetts to California over 24 hours. The loss rate differences vary above and below zero, indicating larger packet sizes

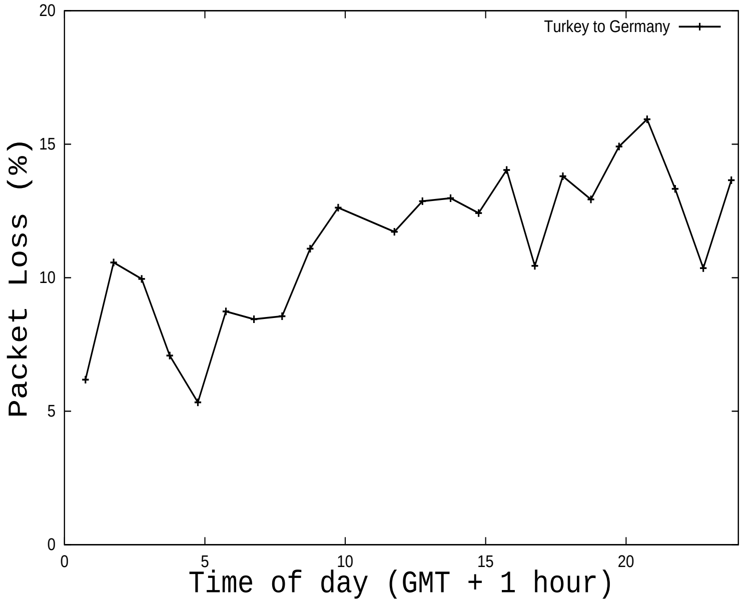


Figure 3.3: Loss over a 24-hour Period

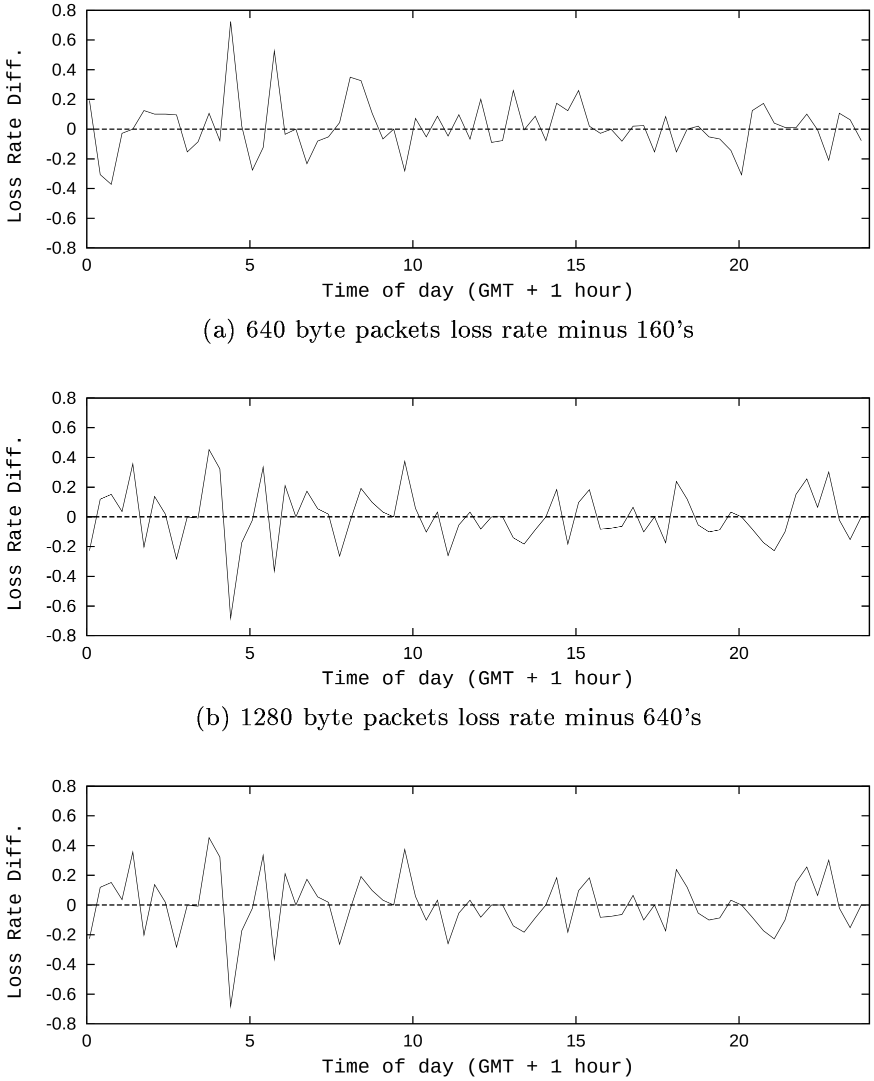
can give higher or lower loss.

Consecutive Loss

In 1995, J. Bolot, H. Crépin and A. V. Garcia measured packet loss and reported that the number of consecutive loss was small if the load of the network was not high [11, 12]. Based on this, they developed a Forward Error Correction (FEC) model, in which redundant information is transmitted along with the original information so that the lost original data can be recovered, at least in part, from the redundant information. FEC depends much on the characteristics of the loss process in the network [13].

- FEC is more effective when the number of consecutive losses are small.
- Sending additional redundant information increases the probability of recovering lost packets, however it uses more bandwidth and introduces extra delay. Potentially it can also increase loss. Therefore, a FEC scheme can be used together with a rate control scheme.
- The amount of redundant information used at any time should depend on the loss process at that time.

There are different FEC algorithms. If the loss rate is not very high, one can choose to send one redundant packet for every n packets. This saves bandwidth, however if the loss rate is high, one can choose sending each audio packet plus a compressed audio packet, e.g. GSM, for some recovery of the signal, this is the method RAT [14] uses.



(a) 640 byte packets loss rate minus 160's

(b) 1280 byte packets loss rate minus 640's

(c) 1280 byte packets loss rate minus 160's

Figure 3.4: Loss Rate Differences for 160, 640, and 1280 byte packets

We have observed that consecutive losses are small for most sessions. Single packet losses are the most common. Consecutive loss of one to three packets happen much more frequently. Figure 3.1.4 shows the occurrences of consecutive loss in all our measurements, note the y-axis is a log scale. We only show one to 50 consecutive losses (corresponding to 1 second), since consecutive losses longer than 50 packets occurs seldom. In many of these cases we have identified some problem with the sender or receiver and the session did not recover after such a long loss sequence. Nevertheless the number of long losses is small compared to the 25,000 we have collected.

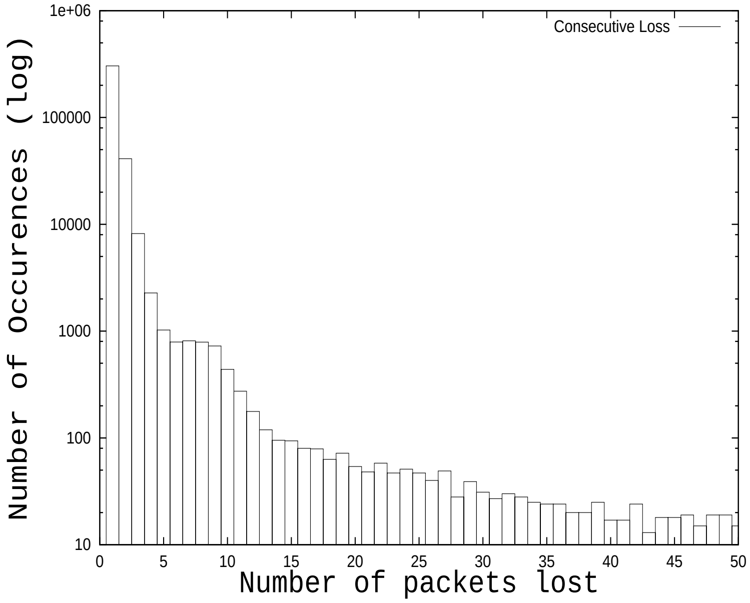


Figure 3.5: Consecutive Loss Histogram

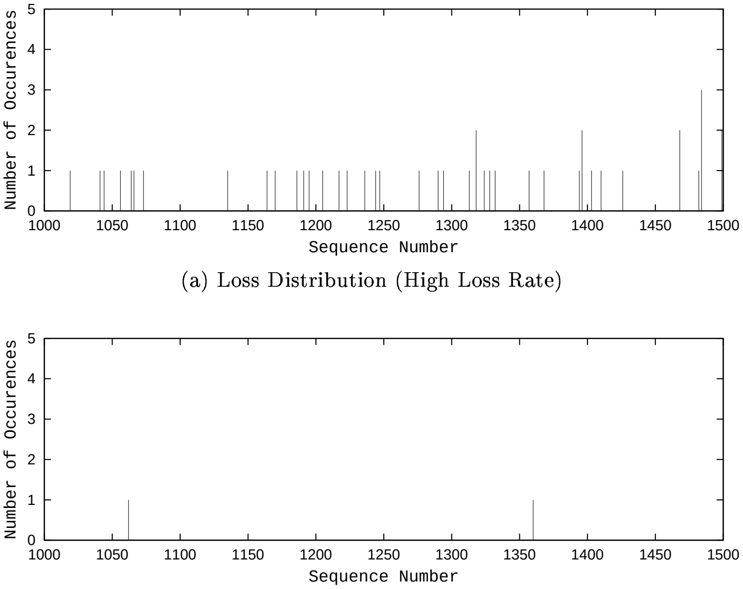
Consecutive Packets

As stated, to use FEC for packet loss recovery, the loss distribution information is useful. If the consecutive losses occur too frequently in a certain part of the conversation, it is difficult to recover the losses even if the *total* loss is low. We found in our measurements that the consecutive losses are normally spread out during the conversation. Figure 3.6 (a) is a typical poor connection with a high loss rate 8.24% (Massachusetts to Turkey). In this transmission, losses occur frequently but are fairly evenly distributed. To show this more clearly, we show the x-axis for the sequence 1000 to 1500. In contrast, Figure 3.6 (b) shows the loss distribution for a much better connection 0.21% (Michigan to California).

For good connections, the length of consecutive packets are longer than the poor ones, which means that more packets are arrived before any loss occurs. Figure 3.7 shows this. We retain ourselves to sequences within talkspurt, as we can't suppose that packets won't be lost in the silent period.

3.1.5 Summary

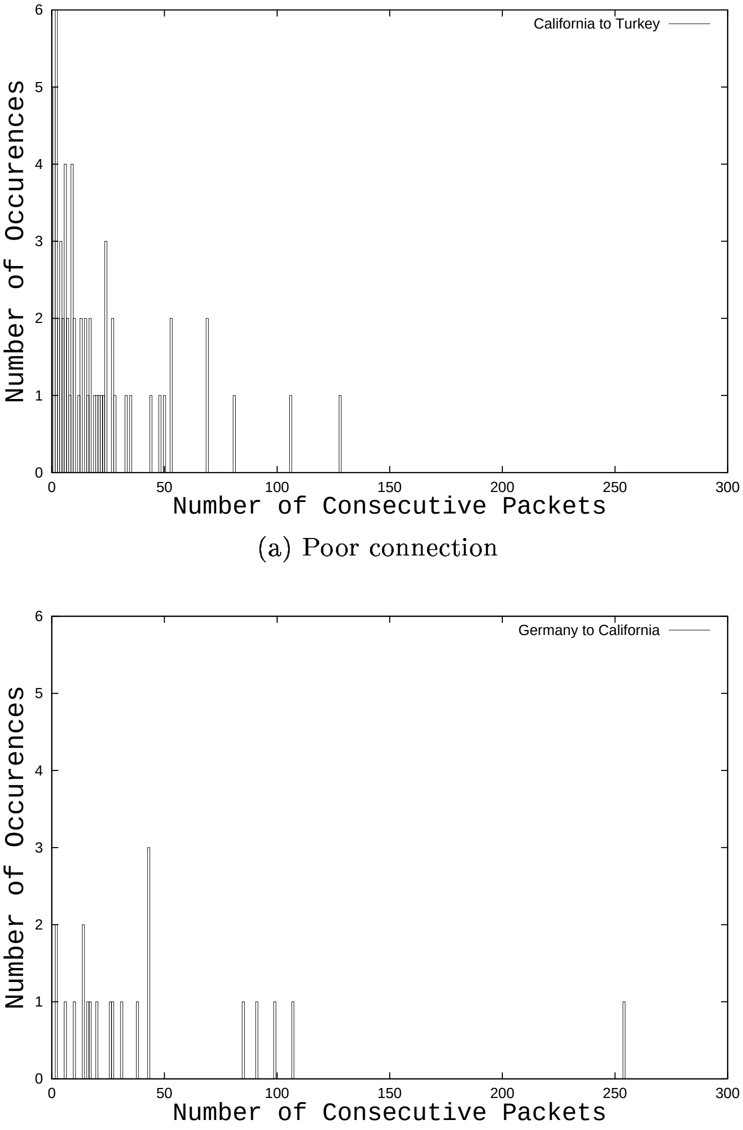
We have taken 15 weeks of measurements so far and have collected nearly 25,000 sample sessions. In most of the transmissions, the loss rate is very low, below 1%, and the number of consecutive loss is also low, usually two to four. So we can conclude that the quality is very good as far as loss is concerned. Packet loss is influenced by the time of day and the day of week, but packet size doesn't have any obvious effect on the loss rate. The losses are usually distributed through the conversations evenly. Therefore most sessions would benefit by using FEC. All of the hosts we used were in academic institutions, the picture might be different for commercial networks, particularly the time of day effect.



(a) Loss Distribution (High Loss Rate)

(b) Loss Distribution (Low Loss Rate)

Figure 3.6: Distribution of Lost Packets



(a) Poor connection

(b) Good connection

Figure 3.7: Successive Packets

3.2 Delay

3.2.1 Definition of Delay

The end-to-end delay is the time taken for a packet to make its way through a network from a source to a receiver. It does not include the delay incurred by the application.

3.2.2 Importance of Delay

- Delay is one indication of quality. Large delays make conversation unnatural and annoying. The International Telecommunications Union (ITU-T) standard G.114 states that a one-way delay should not exceed 150 ms for acceptable quality phone calls [15].

- We mentioned in Section 3.1.3 that delay is one of the four factors in the E-model.

3.2.3 ICMP Delay Vs RTCP Reports

We have looked at both the ICMP echo request and reply times and the RTCP protocol for measuring delay. Both measure the round-trip time, but we are interested in just the one-way delay. Complete accuracy is not our goal in this part of the work, if we can calculate the time to $\pm(10 - 20ms)$ we can see if the delay incurred between the sites lies within the ITU-T standard.

ICMP times are derived from `ping` measurements and using the RTCP sender and receiver reports implemented in using Sicsophone. We sent ICMP packets of 160 bytes as well as used the RTP stream of Sicsophone to measure the round-trip delay. To obtain the one-way delay, we divide the round trip time by two. The question arises if asymmetry can give vastly different delays for the two directions. From our results, this is not the case.

In Table 3.2 the diagonal separates the direction of any two hosts and the delays are not significantly different in either direction. The tests were run over a number of weeks. The top number is the one way delay derived by RTCP reports, while the lower one is the one-way ICMP echo request and reply time. The number in brackets are the standard deviations.

3.2.4 Summary

By looking at the delays for all the connections, we observed that:

- One way delay for most connections is acceptable (less than 150 ms). The connections with Argentina are around this value but delay to Turkey can be as high as 430 ms, so VoIP sessions with this particular host will result in high delays.

- Delay is asymmetric. For each pair of hosts, the delay in one direction is usually not the same in the opposite direction. Our measurements show that the delay differences for both directions are usually small, except for the Turkish host where delay is large and often quite different (110+ ms is common). Section 3.4.2 has further results on asymmetry.

- The delay reported by ICMP and RTCP are very close.

receiver sender	Mass.	Mich.	Cali.	Belg.	Finl.	Swed.	Germ.	Turk.	Arg.	NZ
Mass.	/	38.02 (17.14) ——— 37.54 (16.88)	54.15 (15.81) ——— 55.80 (18.90)	67.11 (15.47) ——— 67.15 (14.70)	97.09 (2.60) ——— 98.00 (4.35)	99.46 (8.51) ——— 98.90 (5.67)	58.39 (5.00) ——— 58.29 (5.08)	388.2 (43.17) ——— 415.5 (61.69)	NA ——— 99.64 (4.91)	149.0 (61.99) ——— 159.8 (59.38)
Mich.	36.44 (15.36) ——— 34.55 (12.73)	/	40.38 (4.50) ——— 40.73 (4.30)	63.54 (4.24) ——— 63.51 (4.44)	88.17 (7.98) ——— 86.83 (2.40)	86.73 (4.71) ——— 85.61 (2.13)	63.61 (8.20) ——— 60.95 (3.71)	358.9 (44.94) ——— 390.9 (38.95)	NA ——— 112.1 (10.55)	124.3 (52.14) ——— 111.9 (8.60)
Cali.	54.45 (16.67) ——— 55.21 (18.57)	40.58 (5.07) ——— 41.35 (4.81)	/	81.04 (2.24) ——— 83.84 (0.49)	105.9 (2.98) ——— 108.3 (1.84)	107.5 (2.39) ——— 107.7 (2.47)	81.53 (1.80) ——— 82.07 (0.65)	386.9 (60.46) ——— 419.2 (35.43)	NA ——— 123.9 (12.44)	81.90 (9.74) ——— 81.24 (2.87)
Belg.	65.24 (10.13) ——— 62.17 (10.13)	63.37 (3.34) ——— 60.68 (3.22)	84.01 (1.33) ——— 81.53 (1.53)	/	31.32 (0.60) ——— NA	33.41 (0.17) ——— 30.23 (0.17)	16.58 (10.36) ——— NA	341.1 (24.61) ——— 337.0 (15.14)	NA ——— 136.5 (7.12)	152.1 (4.30) ——— NA
Finl.	97.84 (4.23) ——— 97.71 (2.81)	86.75 (1.86) ——— 86.55 (2.01)	109.9 (4.70) ——— 109.6 (2.84)	NA ——— 30.72 (0.32)	/	13.62 (1.00) ——— 13.44 (0.65)	26.72 (7.27) ——— 24.07 (5.63)	321.2 (39.32) ——— 309.4 (19.67)	NA ——— 161.5 (12.23)	168.2 (8.68) ——— 170.0 (6.20)
Swed.	99.28 (8.76) ——— 97.77 (5.74)	84.91 (1.94) ——— 84.72 (0.26)	105.6 (2.11) ——— 106.5 (0.76)	33.28 (0.39) ——— 32.75 (0.29)	13.49 (0.53) ——— 13.00 (0.00)	/	29.77 (12.72) ——— 23.50 (0.00)	322.2 (30.34) ——— 333.6 (23.08)	NA ——— 165.6 (17.90)	169.9 (13.08) ——— 176.7 (16.23)
Germ.	63.47 (9.57) ——— 62.39 (9.57)	60.35 (0.54) ——— 60.29 (1.03)	84.40 (9.97) ——— 82.40 (1.42)	NA ——— 11.07 (0.24)	27.75 (7.33) ——— 27.50 (7.23)	29.24 (7.60) ——— 25.87 (4.75)	/	300.7 (39.66) ——— 318.5 (66.39)	NA ——— 149.8 (15.68)	158.7 (15.87) ——— 159.3 11.74
Turk.	379.1 (47.09) ——— 380.6 (26.30)	387.9 (35.46) ——— 379.8 (28.80)	410.9 (43.89) ——— 425.1 (32.50)	330.2 (28.62) ——— 334.1 (19.89)	318.9 (42.44 ——— 332.1 (25.10)	311.1 (8.28) ——— 322.1 (9.38)	378.2 (49.26) ——— 369.2 (37.13)	/	NA ——— 490.8 (25.80)	486.5 (43.40) ——— 473.8 38.21
Arg .	NA ——— 117.0 (30.77)	NA ——— 146.7 (44.18)	NA ——— 152.0 (47.75)	NA ——— NA	NA ——— 164.1 (27.20)	NA ——— 160.9 (47.65)	NA ——— 180.5 (50.36)	NA ——— NA	/	NA ——— NA
NZ .	166.6 (117.7) ——— 156.3 (113.0)	110.4 (15.31) ——— 111.4 (6.69)	80.01 (6.03) ——— 80.95 (3.54)	NA ——— NA	162.8 (7.11) ——— 167.1 (3.35)	173.9 (5.71) ——— 172.1 (4.29)	153.4 (6.67) ——— 153.3 (1.73)	463.1 (69.13) ——— 482.3 (30.62)	NA ——— 206.4 (10.22)	/

Table 3.2: ICMP Echo Request and Reply Times vs RTCP Reports (ms ± std. dev.)

3.3 Jitter

3.3.1 Definition of Jitter

Jitter is the statistical variance of the packet interarrival time. In the RFC1889 [5] jitter is defined to be the mean deviation (smoothed absolute value) of the packet spacing change between the sender and the receiver. If packet i is sent from the sender with timestamp S_i and arrives at receiver at time R_i, then for two packets i and j, the difference of the packet spacing D may be expressed as:

$$D = (R_j - S_j) - (R_i - S_i) = (R_j - R_i) - (S_j - S_i).$$

Sicsophone sends packets of the same size at each interval which means that $S_j - S_i$ is constant.

The difference of the packet spacing D is used for calculating the interarrival jitter. According to RFC, the interarrival jitter should be calculated continuously as each packet i is received. Using D for one particular packet the interarrival jitter J_{i-1} for the previous packet $i-1$ can be calculated according to the formula:

$$J_i = J_{i-1} + (|D(i-1,i)| - J_{i-1})/16.$$

This algorithm is the optimal first-order estimator and the parameter 1/16 gives a good noise reduction (according to the RFC).

3.3.2 Importance of Jitter

- Jitter has an effect on the total delay as it causes delay in the receiver. Packets transmitted at equal intervals from the sender may arrive at the receiver at irregular intervals. A jitter buffer is therefore introduced to compensate for the variance in the delay to smooth the voice stream. A jitter buffer introduces extra delay as it holds the incoming packets for a specified amount of time before they are delivered to the application. There is a trade-off between jitter buffer size and the delay. A small jitter buffer causes less delay but increases the packet loss. A large jitter buffer allows for more variation, but increases delay.

- Similarly with loss and delay, jitter is another parameter when calculating the quality using the E-model. (Sections 3.1.3 and 3.2.2).

3.3.3 Jitter Results

Sicsophone implements jitter calculation in the sender and receiver reports as defined in RFC1889 . However, the interarrival jitter field in the reports are only snapshots of the jitter at the time of the reports [5]. To show the jitter values more precisely, we have calculated the interarrival jitter for each packet. This gives us fine grain control over what we can, as explained previously in the case for loss.

We compare the different methods in Table 3.3 using a sample trace from Michigan to Finland on May 18, 2002 taken at 19:05. From this table we can see that when calculating jitter, using the RTCP reports from Sicsophone loses some precision. We calculate the jitter from the RTP traces of each packet instead.

Method	Average&Dev. (ms)	Max (ms)	Min (ms)
Sics*o*phone (RTCP)	3.13 ± 0.0	3.13	3.13
Jitter (RTP)	3.49 ± 0.71	10.94	0.11

Table 3.3: Comparing RTP and RTCP for Jitter Measurements

Our investigations show that:

- Most connections have low jitter values. The average jitter of all the traces is 3.66 ms. 17.5% of the sessions completed with a jitter value less than 1 ms, and 38.8% less than 2 ms. For all the connections, jitter is usually lower than 4 ms, except for the Turkish and Argentinian hosts.

- Jitter values vary with connections. The jitter differences for connections within US and within Europe are small, and between US and Europe are also small, except the Turkish host. The connections with Turkey have significantly higher jitter (7.46 ms), also the connection *to* Argentina (16.24 ms).

- Jitter is asymmetric. For a pair of hosts, jitter in one direction is usually different from the other. As with delay, the differences are usually small, except for the Turkish and Argentinian hosts. For the Turkish host the difference was measured as 3.06 ms, whilst for the Argentinian host the difference is as high as 12.23 ms. with average jitter *to* it 16.24 ms and *from* it 4.01 ms.

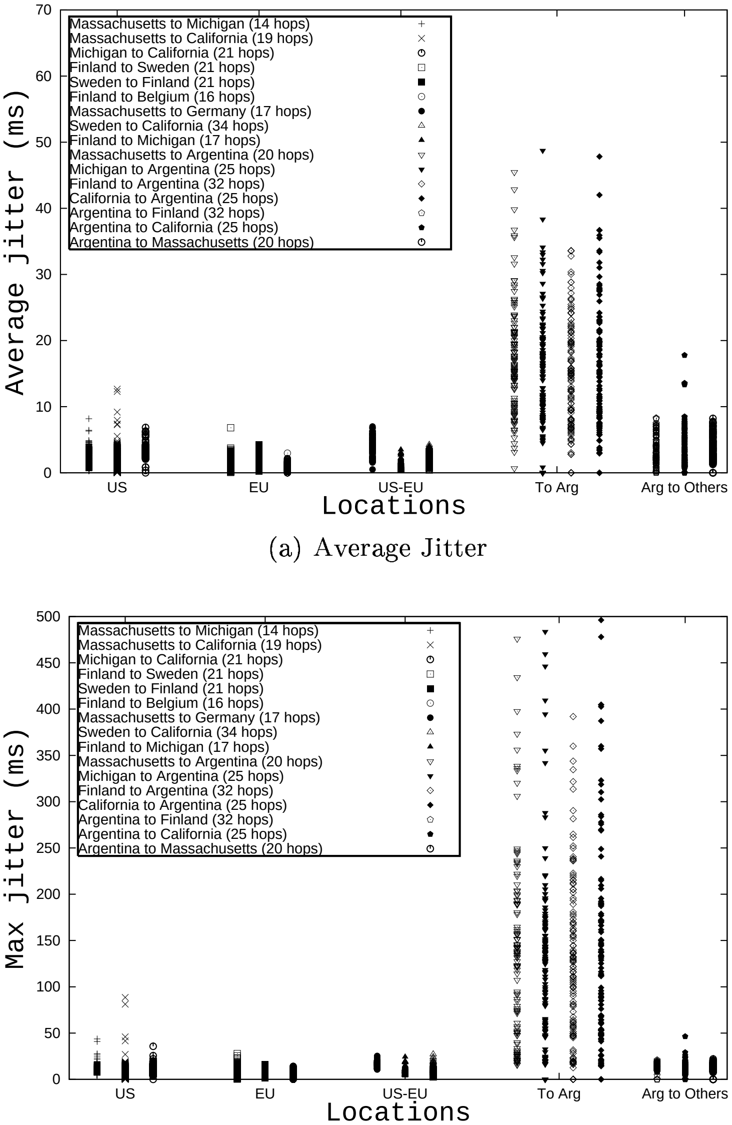
Figure 3.8 (a & b) plots the average and maximum jitter values for connections grouped into regions. We have chosen US, EU, US to EU and Argentina as geographic groups. We chose this grouping to reflect the difference in jitter *to* Argentina. Both the average and maximum jitter plots show significantly higher jitter to South America. Each 'region' contains approximately 1200 data points, i.e. for every connection we have about 400 jitter measurements.

Figure 3.9 and 3.10 show the jitter for four connections, the first one 3.9 shows only the jitter, the minimum, average and maximum represented by the three columns. The maximum jitter is quite clearly much larger than both the minimum (obviously) but also the average.

However looking at the jitter with regard to the delay shows a slightly different picture, the *ratio* of the delay to the jitter reveals the first (Finland to Sweden) connection has a high delay-jitter ratio.

3.3.4 Summary

In our tests, the jitter values are low for most of the connections. Jitter of less than 1 millisecond is optimal for VoIP traffic [16]. Therefore, we can conclude that in general the quality is excellent in terms of jitter, but not all over the world, such as to Argentina (16.24 ms). The connections to it will experience quality problems.



(a) Average Jitter

(b) Maximum Jitter

Figure 3.8: Average and Maximum Jitter

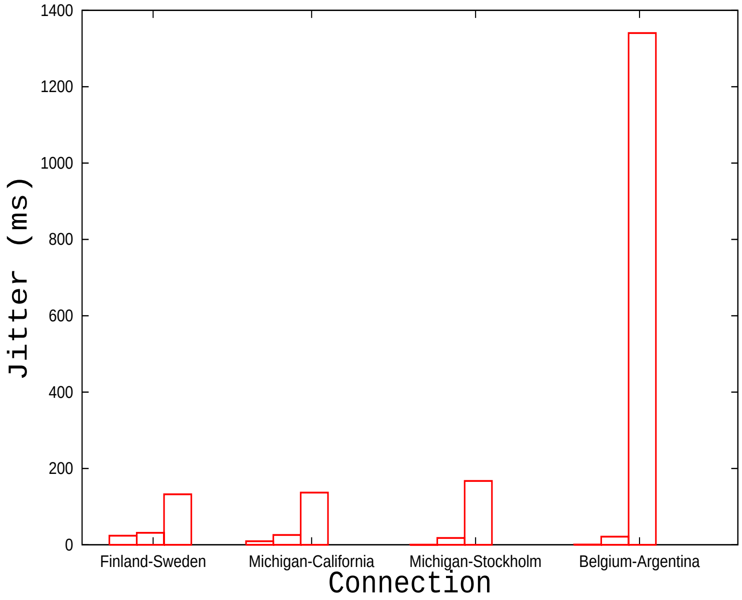


Figure 3.9: Minimum, Average and Maximum Jitter

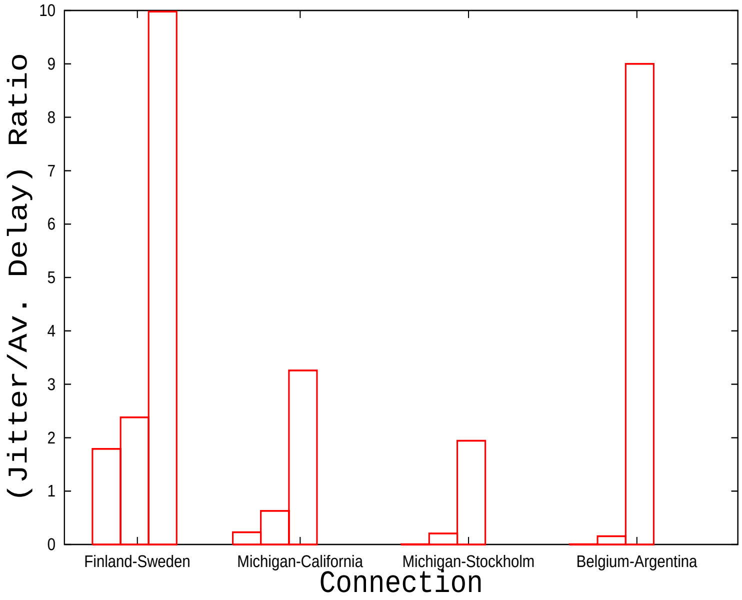


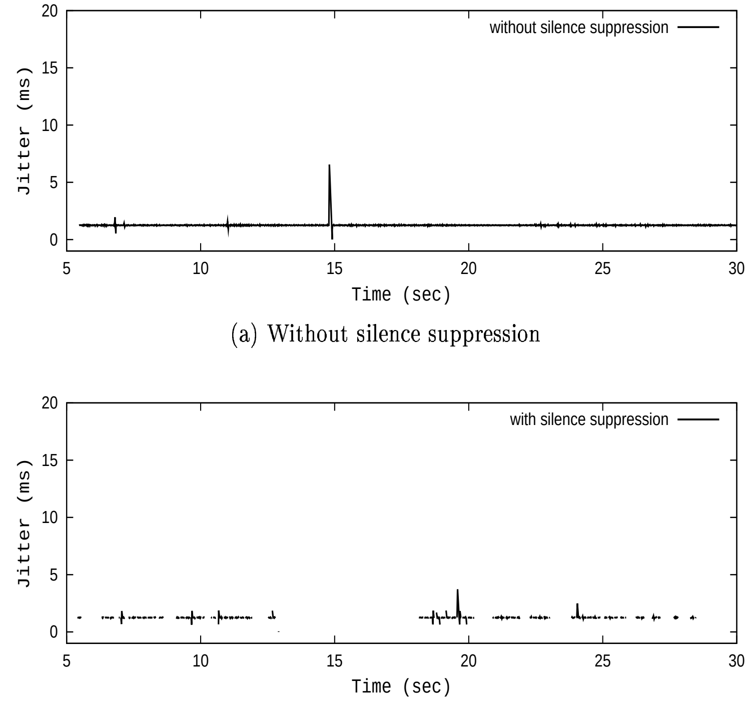
Figure 3.10: Jitter Delay Ratio

3.4 Network Considerations

3.4.1 Silence Suppression

Using silence suppression means no packets are sent when the sender is silent; without silence suppression packets are sent even during silence. Packet are sent at regular intervals but the payload is simply 0's. In a typical interactive dialog, each speaker listens for about half the time, thus it is unnecessary to transmit packets during the speaker's silence. The main problem is detecting when the speaker is actually silent, background noise and sudden sounds can reactivate the sound level past a threshold so packets begin to be sent. This area of human-computer interaction is known as Voice Activation Detection (VAD).

We have tested our links using both silence suppression and not. We found that using silence suppression did not have any significant influence on the network load, i.e. we could not cause any extra loss by not using suppression. Figure 3.11 plots the jitter values for a connection from Massachusetts to Michigan. With silence suppression the average jitter is 1.250 ms, while without the average is 1.281 ms. The difference is small. In total we performed 1250 silence suppression tests.



(a) Without silence suppression

(b) With silence suppression

Figure 3.11: Comparison of With and Without Silence Suppression

In high load situations (or low capacity) silence suppression may influence the network, a modem connection for example. Without silence suppression more data is sent so that we can get more sampling of the network, but the network is slightly more loaded. For good connections where the loss is very low one can choose to send without silence suppression. For bad connections such as to Argentina and Turkey, it is advisable to use silence suppression to reduce the load on the network, which may result in better quality. From our measurements we could not find two simultaneous plots which depicts this clearly though.

3.4.2 Network Asymmetry

By network asymmetry we mean the path from one site to a peer is not the same in the reverse direction. More specifically, the route across the Internet does not traverse the same routers interfaces. This is quite normal as one site may have an agreement with one ISP whereas in the reverse direction the site maybe use another ISP. Potentially this can result in differing loss rates, delay and jitter for both data and VoIP over the Internet [17]. This is not as big problem for data transmissions as for VoIP, if a Web transfer takes slightly longer in one direction then most people will not notice (and don't care). For VoIP network asymmetry is more important. Network asymmetry could potentially result in different perceived quality for the parties.

Our results show, although asymmetry exists it is not so serious in terms of VoIP quality as we had expected. In our 3750 bidirectional tests, the loss, delay and jitter differences were usually small. For example, the loss values were up to 0.5-1.25% in opposite directions. Figure 3.12 is an example of route asymmetry where loss is concerned.

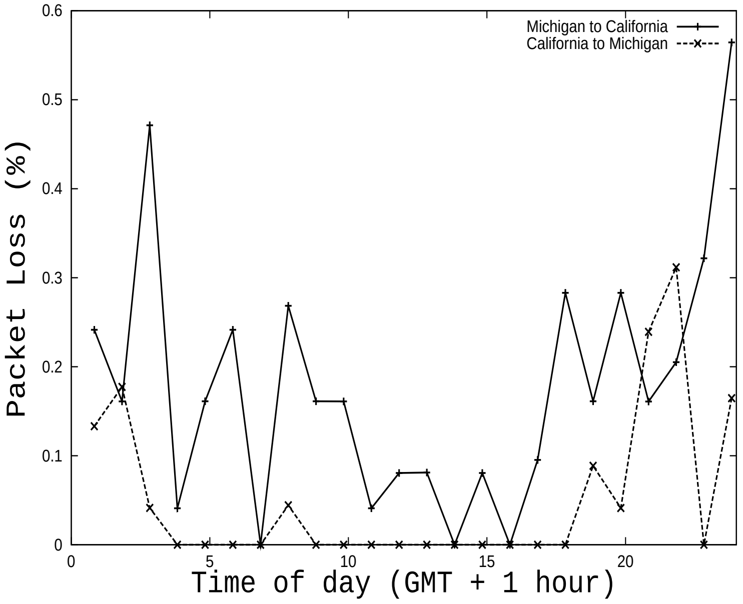


Figure 3.12: Route Asymmetry

Asymmetry also exists in the number of hops between a pair of hosts. The route from one host to another can be differ from the reverse direction. The number of hops can be different, this was shown in Table 2.2, Chapter 2.

3.4.3 Other Measurement Tools

Tools like pchar [18] and Sprobe [19] can be used for inferring more details about particular path characteristics. Pathchar gives us information about the bandwidth, delay and loss of links along an end-to-end path. One problem with pathchar is that it can take a long time to complete, especially in the case of a connection with many links. Additionally it can place heavy load on the links themselves. Another problem is it assumes that the link bandwidths are symmetric. Sprobe is one alternative for estimating the bottleneck bandwidth for links but is not designed for VoIP use.

3.5 Summary and Quality Comparison with 1999

In this chapter we have looked at the loss, delay and jitter parameters of VoIP quality. Similar measurements were done in 1999 using Sicsophone. The hosts that are available to us both in 1999 and in 2002 are shown in Figure 3.13. The measurements in 1999 showed that most calls had a loss rate less than 5%, and delays less than 150 ms. This result still holds in 2002. The overall quality has improved despite the much heavy use of the Internet. Table 3.4 compares the results for the measurements. The values in the brackets are standard deviations.

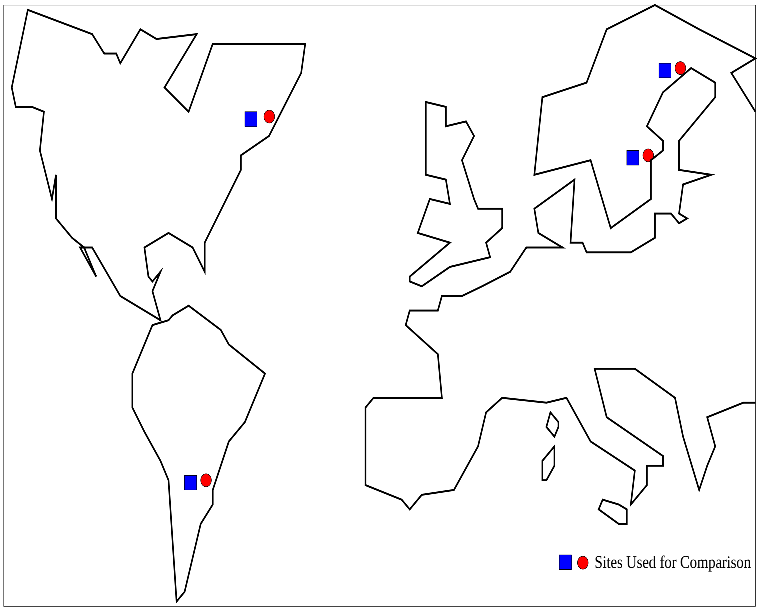


Figure 3.13: Sites Used in 1999 and 2002

Quality	1999	2002	%Diff.(+/-)
Jitter	45.1ms	22.6ms(13.7)	-50.0%
Loss	1.2%	0.5%	-58.3%
Delay	115ms	84.95ms(44.85)	-26.1%

Table 3.4: 1999 and 2002 VoIP Quality Differences

Chapter 4

Related Work

In this chapter we consider work done by other researchers and compare their results with ours.

4.1 Measure Quality

In 1997, Maxemchuk and Lo measured the quality of intra state, inter country, and international Internet links [7]. They defined the quality of a connection as the “quality fraction of time that the signal is received without distortion for intervals of time that are long enough to convey active speech segments”. The conclusions were that the Internet was capable of carrying voice data although there were some difficulties. They had two important observations of how the length of connection and the time of day affected the quality of a connection:

- Longer connections gave worse quality
- The time of day affected quality significantly. The quality was much worse during busy hours

The same conclusions were drawn by Dong Lin in 1999 [20]. She measured the end-to-end delay, packet loss and jitter of both US and international links, and concluded that packet size did not influence packet loss. Our conclusion agrees with hers even though our measurements were taken three years after hers. She also observed that the length of consecutive losses is small. In her tests, loss lengths of one and two packets were predominant, while in ours, it is one to three.

Compared with her results, we observed lower delays, however since we did not have access to exactly the same links as she did we cannot say this categorically. Even our connections which are “longer” give lower delay but identical test should be conducted to verify this. In her tests three years ago, the round trip delay was under 300 ms (one way delay 150 ms) for only the hosts inside US, while in our tests most of the links, except the Argentinian and Turkish host, have round-trip delay under 300 ms.

Both of these works use UDP for measurements with the topology being centralized. We use both RTCP and RTP for our measurements and we use a full-mesh topology. A full-mesh topology gives the best possibilities for testing the intervening links. Moreover, their measurements lasted only a few days, whilst ours more than three months.

The E-Model, defined in the ITU-T Rec. G.107 [21], is a network-planning tool and can be used to estimate impairments in voice quality. R. G. Cole and J. H. Rosenbluth used the E-Model to measure the quality of VoIP and showed that voice quality was not only affected by delay, loss, and jitter, but also the coder frame size [8]. They characterized these four factors into a single error mask, the R-factor, which results in a single value for the quality. We did not take our work to this level, but have all the information needed to do so. To derive the R-factor of the E-model we can use the delay, loss and jitter. We use 8bit PCM as coding with a frame size of 0.125 ms (coding speed 64 Kbit/s).

4.2 Measuring Delay

Some work focus on specific characteristics of voice transmission. One of the early works of delay measurement was done by Bert Dempsey, Matthew Lucas and Alfred Weaver at the University of Virginia in 1994. They studied the round-trip delay of the University's campus network, a Local Area Network [10]. They sent 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 datagrams, the remote machine immediately transmitted a datagram of the same size and the same sequence number. The delay was calculated as the difference of the timestamps. They measured three connections in the campus network of different length and concluded at busy times of the day the delay was larger.

4.3 Measuring Packet Loss

In 1995, Jean-Chrysostome Bolot, Hugues Crépin and Andres V. Garcia concentrated on the measurement of packet loss and found that usually the number of consecutive packet loss was small if the load of the network was not high [11, 12]. Based on this observation, they developed a model indicating that forward error correction, FEC, could rectify most situations where packet loss is a problem.

4.4 Other Work

4.4.1 Correlation between Delay and Packet Loss

In 1998, Sue Moon, Jim Kurose, Paul Skelly and Don Towsley at the University of Massachusetts performed measurements for investigating the correlation of delay and packet loss [22]. Their conclusion was that delay and loss were closely related to each other: the receiver could have taken the increased delay as an indication of likely future packet loss; the decreasing delay following loss indicated a future uncongested period of time. We have not studied this phenomena.

In 1999, Wenyu Jiang and Henning Schulzrinne identified different aspects of delay and loss, i.e. the correlation between loss and delay as well. More specifically how the previous delay or loss affects the delay or loss probability of the next packet [23]. By analyzing delay and loss models, they concluded that delay and loss have strong correlation: they depend on their previous values.

4.4.2 Routing Asymmetry

In 1996, Vern Paxson investigated routing behavior in the Internet [9, 10]. He made an analysis of 40,000 end-to-end Internet route measurements by using traceroute and a custom tool between a diverse collection of Internet sites. He analyzed the impact of routing asymmetry to the network: it affects the network, for example, the estimation of one-way delay because one-way delay is based on half of the round-trip time. It also makes it difficult to locate problems in the network because the problem may exist only in one direction. Paxson's measurements show that it is very likely that a path through the Internet is more or less different from the path in the opposite direction. Our measurements verify this: the routing asymmetry is very common but does not adversely affect the loss, delay and jitter.

Chapter 5

Open Issues and Future Work

The following is a list of suggestions if this work is to be continued:

- More hosts can be added for more tests.
- Send audio packets continuously for more sampling. In our measurements we usually send once an hour. More sampling will show the quality of the network more exactly.
- Add different call lengths. currently our sample call lasts about 150 seconds (for 160 bytes/packet). We could try short, medium and long duration calls.
- Include signaling of the calls.
- Investigate the reliability of VoIP, that is the percentage of the calls that are completed. A lot of work has been done to measure the quality of VoIP. Little attention has been paid to the reliability, but reliability is the user's highest concern. Users care more about if the calls are successful than quality issues. In the context of our work some definition of reliability could be used, if more than X seconds of audio are lost what is the reliability? How many calls were affected by a loss of more than ten packets more than once for example. Another possible starting point would be to find out what annoys and irritates users of PCM audio to make them terminate the connection and apply that to our measurements. We would have some idea how many of the calls were "reliable" from a user perspective.
- The log files can be used for further research. We have collected almost 25,000 log files, and we used them for looking at packet loss, delay and jitter. They can be used for other investigations.
- Use the measurement infra-structure on networks connected to commercial networks. This might give a different pictures for the usage over 24 hours. We would expect much more traffic in the evenings and nights and less during the daytime.
- More investigations on the connectivity of hosts. We have discovered at least one host where traffic shaping is done, this is useful information when looking for explanations of observed effects.

- Try different codings. In our measurements we only used 8 bit PCM.
- Calculate a matrix of the quality using the E-model.

Chapter 6

Lessons Learned

Performing VoIP measurements on such a large scale is a complex task. We have obtained some experience which will be useful if we were do such measurements again in the future.

- Plan carefully, including what to measure, the measurement method and how to obtain the results. A systematic plan at the beginning is very important. The plan should be as complete and detailed as possible. One example is how to arrange and name the log files.

- Make the infra-structure as robust as possible to handle unexpected events. For example if the receiver is started, but the connection to the sender fails, the program should stop the receiver, then exit immediately. Possibly re-starting it later.

- Pay attention to the load of certain machines when performing simultaneous tests. We observed problems when testing the effect of packet size and silence suppression when calls must be started at the same time. The load on the machines is low for sending and receiving packets, but machines capability of handling several ssh connections at the same time was limited. Some of our machines can handle at most 4 sessions at the same time. We can only conclude other people were using the machines or they were very slow (not only the CPU, but the disk since the sample conversation is 0.5 Megabyte).

- Be prepared for outages and changes. Configurations change on remote machines, reboots happen, etc. Keep active contact with the remote system administrators is crucial.

Chapter 7

Conclusions

We have investigated the quality of telephone calls on the Internet with the focus on quality characteristics of loss, delay and jitter. Our tests lasted 15 weeks and we have collected almost 25,000 log files.

Sicsophone was used as the central tool in the measurement because it offers almost full VoIP capabilities with comprehensive logging features. We have added functionality to Sicsophone through UNIX "wrappers" to enhance the functionality, in particular handling error situations, collating logged data and bi-directional call scenarios.

More test sites have been added since the measurements were last done (ten instead of 5) and more importantly connected them in full-mesh topology to obtain many more measurements. Also the sessions are started automatically via UNIX scheduling enabling us to make more detailed 24 hour measurements.

During our experiments, we have looked if packet size and silence suppression affect the measurements. We conclude they do not. We also looked at network asymmetry. It does exist from our continual measurements, but the effect on VoIP sessions is less than we expected.

We conclude that Voice over IP is still feasible in 2002, although not in a global scale. Our investigations show that quality varies with time and connection. In general, most of our calls exhibited good quality. 66% calls completed with zero loss and 98.2% finished with a loss rate less than 10%. Delay is acceptable for most of the links, less than 150 ms, and jitter is also low with the average value for all links 3.66 ms. We saw that VoIP is possible with good quality within the US and EU and between the US/EU but not outside these areas. The conclusions agree with the general comment made by Henning Schulzrinne [24].

We further conclude that the quality of Voice over IP has improved since 1999. Compare with the results in 1999, the loss rate is reduced by 50%. The delay and jitter are also improved. Although there has been a great expansion of Internet, greater efforts have been put into increasing the bandwidth of the links to improve the capacity and performance for both data and VoIP sessions.

Appendix A

Remote Login with Secure Shell (SSH)

SSH is a program for logging into a remote machine and for executing commands on a remote machine. It provides secure, encrypted communications between two hosts on the network. SSH implements public key authentication. The server knows the user's public key, and only the user has the private key. The encryption method can be either DSA or RSA. As we have many hosts, it's easier if all the keys are of the same type. We have chosen DSA for all the hosts.

The hosts that are available to us either use openSSH or ssh2. The way that they handle the keys are very different.

A.1 OpenSSH Hosts

To remotely login without password, users need to create a public and private key pair with ssh-keygen. There are two versions of OpenSSH. Version two provides better security and is more commonly used now.

A.1.1 SSH version 2

The default place for saving the keys is $HOME/.ssh, and the default names for the private and public key are id_dsa and id_dsa.pub. Then user needs to copy the public key to the remote host and append it to the file $HOME/.ssh/authori zed_keys2. That is the key is saved as plain text in the file authorized_keys2. When a user tries to login, the remote machine checks if the user's key is listed in authorized_keys2. If so, it will send a challenge to the user, a random number encrypted by the user's public key. Only the user has the private key and can decrypt it.

Steps of creating and copying key:

1. On user side, use command ssh-keygen to create a DSA key pair.

   ```
   $ ssh-keygen -t dsa
   ```

ssh-keygen will ask a passphrase for new key. Empty passphrase is also fine. ssh-keygen creates a .ssh directory in user's home directory if it doesn't exist, and stores both the public and the private key there.

2. Copy the public key from user to remote machine. This can be done either on user side or remote machine side.

   One way of copying the key is to use the Secure Copy, scp. It is used for copying files over the network securely.

   If on user side:
   ```
   $ scp $HOME/.ssh/id_dsa.pub remote_user_name@remote_machine_name:$HOME/.ssh
   ```

   If on remote machine side:
   ```
   $ scp user_name@user_machine_name:$HOME/.ssh/id_dsa.pub $HOME/.ssh
   ```

   The key name will remain unchanged on the remote machine. It can be given another name, but not necessary.

3. On the remote machine, append the public key to the file $HOME/.ssh/authorized_keys2

   ```
   $ cd $HOME/.ssh
   $ cat $HOME/id_dsa.pub >> .ssh/authorized_keys2
   ```

   Then the public key file is not useful any more and can be deleted. After this, the user can log into remote machine without password.

A.1.2 SSH version 1

Version 1 and Version 2 are similar to each other. The differences are:

- Use command `ssh-keygen -t rsa1` to generate the keys. It's not possible to generate dsa keys in version 1. The keys are called identity and identity.pub by default.
- The public key should be saved in the file $HOME/.ssh/authorizedkeys.
- Version 1 does not support key conversion by using `ssh-keygen`.

A.2 ssh2 Hosts

Users also need to create a public and private key pair with ssh-keygen. Moreover, users need to create a file called identification, specifying the file name of the private key and save it together with the private key under the directory $HOME/.ssh2. Then user should copy the public key to the remote machine and save it in the directory $HOME/.ssh2 on the remote machine. After this in the same directory on the remote machine let the file called authorization specify the file name of the public key. When the user tries to authenticate himself, the server checks $HOME/.ssh2/authorization for filenames of matching public keys and sends a challenge to the user. The user is authenticated by signing the challenge using the private key.

Steps:

1. On user side, use command ssh-keygen to create the DSA key pairs.

   ```
   $ ssh-keygen -t dsa
   Generating 1024-bit dsa key pair
   9 o.oOo..oOo.o
   Key generated.
   1024-bit dsa, created by user_name@user_host_name Mon
   Apr 08 09:11:02 2002
   Passphrase:
   Again :
   Private key saved to /home/user/.ssh2/id_dsa_1024_a
   Public key saved to /home/user/.ssh2/id_dsa_1024_a.pub
   ```

   ssh-keygen creates a .ssh2 directory in user's home directory if it doesn't exist and stores both the public and the private key there.

2. On user side, let identification file in the .ssh2 directory specify the private key.

   ```
   $ cd $HOME/.ssh2
   echo "IdKey id_dsa_1024_a" > identification
   ```

3. On user side or remote machine side, copy the public key from user to remote machine, save it in the directory $HOME/.ssh2 with a unique name, like user_name.pub. As there can be many public keys saved in this directory, the key name should tell whose key that is.

   If on user side:

   ```
   $ scp $HOME/.ssh2/id_dsa_1024_a.pub remote_user_name@
   remote_machine_name:$HOME/.ssh/user_name.pub
   ```

   If on remote machine side:

   ```
   $ scp user_name@user_machine_name:$HOME/.ssh2/d_dsa_
   1024_a.pub $HOME/.ssh2/user_name.pub
   ```

4. On remote machine side, let authorization file specify user's public key.

   ```
   $ cd $HOME/.ssh2
   ```

   If file authorization doesn't exist

   ```
   echo "Key user_name.pub" > authorization
   ```

   If file authorization exists

   ```
   echo "Key user_name.pub" >> authorization
   ```

   After this, the user is allowed to log in to the remote machine without password.

A.3 OpenSSH Hosts to ssh2 Hosts

The keys of OpenSSH and ssh2 look different and should be saved at remote host in different ways. Therefore, the keys of OpenSSH and ssh2 do not recognize each other. OpenSSH version 2 supports key conversion. We can convert an OpenSSH key to log into ssh2 hosts. However, ssh2 doesn't support any conversion, therefore we can only login from a OpenSSH host to a ssh2 host, but not the opposite.

Steps:

1. On user side, Convert the key with the command `ssh-keygen -e -f $HOME/.ssh/id_dsa.pub`
   The key will bedisplayed as text for the user. The original key remains unchanged.

2. Save this plain text in a file with the name user_name.pub for example, and copy it to the remote ssh2 host in the directory $HOME/.ssh2. The copy can be done on either side as explained above.

3. On remote host, let authorization file specify user's public key.

   After this, an OpenSSH user can log into remote ssh2 host without password.

When first time a user logs into a remote machine, the user will be told that the authentication cannot be established, and be asked if he wants to continue connecting. By answering 'yes', the user will be permanently added to the remote host's list of known hosts, and is allowed to log in without password. After this the use will be able to login directly.

Bibliography

[1] L. Sweet, "Toss your dimes-internet video phones have arrived," *ZD Internet Magazine*, August 1997.

[2] O. Cornéer, ""bredbandsbolaget slipar sina ip-knivar" from finansvision," June 2002.

[3] D. De Vleeschauwer, J. Janssen, G. H. Petit, and F. Poppe, "Quality bounds for packetized voice transport," *Alcatel Telcommunications Review*, pp. 19–24, 1st Quarter 2000.

[4] O. Hagsand, K. Hanson, and I. Marsh, "Measuring Internet telephony quality: Where are we today?," in *Proceedings of IEEE Conference on Global Communications (GLOBECOM)*, Rio, Brazil, November 1999.

[5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," RFC 1889, Network Working Group, Feb. 1996.

[6] J. Smith and M. Jones, "A technical report on speech packetization," , Working Group on Specialized Signal Processing, 1995.

[7] N. F. Maxemchuk and S. Lo, "Measurement and interpretation of voice traffic on the Internet," , AT&T and Bell labs, 1997.

[8] R. Cole and J. Rosenbluth, "Voice over IP performance monitoring," , AT&T, 2001.

[9] O. Hersent, D. Gurle, and J. P. Petit, *IP Telephony - Packet Based Multimedia Communication Systems.* Addison-Wesley, 2000.

[10] B. Dempsey, M. Lucas, and A. Weaver, "An empirical study of packet voice distribution over a campus-wide network," tech. rep., University of Virginia, March.

[11] J. C. Bolot and H. Crepin, "Analysis of audio packet loss in the Internet," , INRIA, France, 1995.

[12] J. C. Bolot and A. Vega-Garcia, "The case for FEC-based error control for packet in the Internet," , INRIA, France, 1997.

[13] J. C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-based error control for Internet telephony," in *INFOCOM '99*, (New York), March 1999.

[14] Mice Project (RAT: Robust Audio Tool) Multimedia Integrated Conferencin g for Euro-pean Researchers. University College London.

[15] ITU-T Recommendation G.114, "General characteristics of international telephone connections and international telephone circuits: One-way transmission time," February 1998.

[16] www.extremenetworks.com/solutions/case_studies/VoIP_SB.pdf.

[17] V. Paxson, "End-to-end routing behavior in the Internet," in *SIGCOMM Symposium on Communications Architectures and Protocols*, Stanford, California, August 1996.

[18] B. A. Mah, "pchar: A tool for measuring internet path characteristics." http://www.employees.org/ bmah/Software/pchar.

[19] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "Sprobe: Another tool for measuring bottleneck bandwidth," in *Work-In-Progress Report at the Third USENIX Symposium on Internet Technologies and Systems (USITS 2001)*, San Francisco, California, March 2001.

[20] D. Lin, "Real-time voice transmissions over the Internet," , University of Illinois, Urbana-Champaign, 1999.

[21] "The E-Model, a computational model for use in transmission planning." ITU-T Recommendation G.107, 1998.

[22] S. B. Moon, J. Kurose, P. Skelly, and D. Towsley, "Correlation of packet delay and loss in the Internet," , University of Massachusetts, Amherst, January 1998.

[23] W. Jiang and H. Schulzrinne, "QoS measurement of Internet real-time multimedia services," , Columbia University, December 1999.

[24] H. Schulzrinne, "What is difficult about replacing the telephone network?." RVK keynote, June 2002.