A real-time tool to display the quality of voice communications over IEEE 802.11b networks

Assignment for the course 2G1325:
Practical Voice Over IP (VoIP): SIP and related protocols

Royal Institute of Technology (KTH), Stockholm

Last version of this document: 28th June 2004

by

Victor Yuri D. Nunes

Abstract

We have developed an utility for Sphone, a custom VoIP tool, to display an estimated quality of a voice communication in real time. This utility also provides a log file to create statistics and analyze the parameters that affect the quality of voice communication, namely round-trip time (RTT), jitter and loss. Moreover, we have performed experiments with our utility and we present some results and conclusions of these measurements. These experiments have been conducted over an 802.11b wireless LAN to get an insight on to how well these networks are suited for voice communication.

Acknowledgments

I would like to thank Juan Carlos Martin Severiano for all his support and motivation, thank you Juan. I would like to thank Ian Marsh (my thesis advisor) for carefully reviewing this document. To my dear girlfriend, in Portuguese, “Iarah amo-te muito”.

Table of Contents

List of tables

List of figures

List of terms and acronyms

AP	Access Point
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IMIT	Institute of Microelectronics and Information Technology
IP	Internet Protocol
ISP	Internet Service Provider
ITU	International Telecommunication Union
KTH	Royal Institute of Technology
LCN	Laboratory of Computer Networking
LCC	Logic Link Control
MAC	Media Access control
MOS	Mean Opinion Score
PLCP	Physical Layer Convergence Procedure
PMD	Physical Medium Dependent
OSI	Open System Interconnect
QoS	Quality of Service
RR	Receiver Report
RTT	Round Trip Time
SIP	Session Initiation Protocol
SR	Sender Report
TCP	Transmission Control Protocol
VoIP	Voice-over-Internet Protocol
WLAN	Wireless Local Area Networks

Chapter 1 Introduction

1.1 Overview

Voice over IP (VoIP) is already a reality in business, institutional and home networks. It has proved to be a serious competitor to the public switched telephone network, in terms of cost, efficiency, quality, versatility, and reliability. It started with small applications for private consumers to make calls from a computer to computer. Then it progressed from computer to a regular phone networks via gateways, overtime it gained acceptance among the business community, as it could provide a whole telephony system over an already available IP network, and for a fraction of the cost. As a result VoIP provides the solution for the long time desired convergence between voice and data networks.

However, in spite of its many advantages there are some limitations. Classic telephony networks are designed to offer predictable quality voice communication by using circuit-switched technology, with dedicated channels for each voice session. VoIP, on the other hand, as it is developed to work over IP networks, shares the drawbacks of this packet-switched technology: there is no guaranteed quality of service. IP networks were not designed for voice but to transmit data, and they are based on the 'best effort' principle, which means that some packets can be lost on the way, thus the applications using the network are responsible for a reliable transmission of the data. An IP network is a shared resource utilized by different applications and devices that compete for access to the channel, which can lead to bottlenecks and delayed or even lost packets.
For VoIP to work properly some requirements have to be fulfilled, or, in other words there are demands for some Quality of Service (QoS) from the network. The voice stream must not suffer a delay higher than 150 ms (including processing delays added at the end systems plus the network's latency) because this would lessen the interactivity of the conversation. Furthermore, depending of the codec utilised to transform the analog voice signal into a digital stream of packets, the percentage of lost packets must be kept under a certain minimum; if this does not happen it could be impossible to reconstruct the voice at the listener in a comprehensible way.
There are mechanisms and technologies that provide the QoS required by VoIP in specific networks, but not all networks support those mechanisms.

Not only has the technology of packet-switched networks mature, thus decreasing the difference in terms of suitability for voice transmission between circuit-switched and packet-switched networks, but improvements in voice processing and concealment techniques make voice communication more robust against poor network conditions. However, as the capacity of the networks increases, it also increases the number of users and the use that they make of the network. New applications are more greedy in terms of traffic bandwidth. This generates both impairments for demanding QoS traffic and the need for monitoring of the network conditions, in order to know if certain applications (like VoIP) can be used with a given minimum quality. This monitoring, in the case of voice communication, it is not an easy task. The quality of a voice call not only depends on network conditions, but also on perceptual characteristics of the end users and how these network impairments affects the user's perception. Also other factor is the user's expectations, since mood and human memory can increase or decrease the score that a user would give to the quality of a particular voice call. Furthermore, two identical sets of errors in the transmission can have different impact on the quality depending on the part of the speech they occur in.

We have developed a small tool that, combines network condition analysis and some theories about the impact of such conditions on human perception, to give an estimate of the quality of a voice call in real-time.

1.2 Report Structure and contents

In Chapter 1 a brief introduction to voice-over-WLAN is presented. Chapter 2 offers a description of the WLAN's layered structure, emphasizing the most relevant protocols for our work. Real time protocols are introduced. Quality of Service metrics are discussed in chapter 3. In chapter 4 a description of Sphone and our tools is presented. Chapters 5 presents experiments we performed using our real time tool. Conclusions and future work are discussed in Chapter 6.

Chapter 2 – Background

In this chapter we present the most relevant concepts on which we base our work.

2.1 802.11b Network Layers and protocols

The 802.11b protocol stack is a simplified version of the OSI model and consists of five layers: application, transport, network, data link and physical layer (figure 2.1)[1].

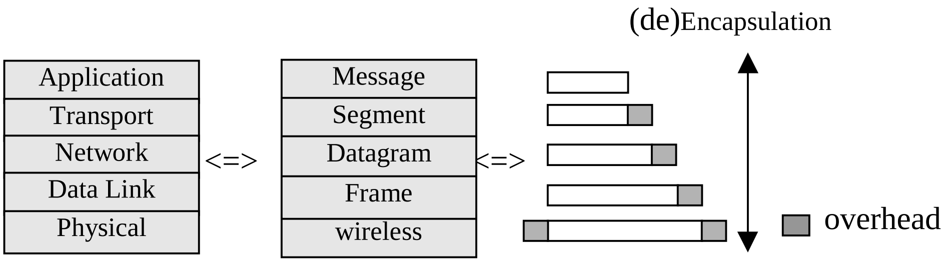

Figure 2.1 Network Layers and overheading

The application, transport and network layers are usually implemented in software, while data link and physical layers are commonly implemented in network interfaces (hardware).
Each layer has its own protocols and the process of passing information between adjacent layers is generally well defined by standard bodies. In figure 2.1 we illustrate how data units are encapsulated and emphasize the packet overhead. Which has a significant impact on voice-over-WLAN QoS because it increases the delay.

We will now describe the 802.11b network layers and protocols used in VoIP communications. Real-time protocols are described in the next section.

2.1.1 UDP protocol - transport layer

The transport layer generally provides transport services to the application layer messages using mostly two protocols, TCP and UDP. TCP offers reliable packet delivery using control mechanisms; while UDP is commonly used when the process needs to send data at a minimum rate, it is the case of IP-Telephony (VoIP). UDP provides an unreliable, connectionless service and adds only an 8 byte UDP header (i.e. only 8 bytes of overhead). Because there is no mechanism to prevent packet losses applications running over UDP usually are loss-tolerant.

<u>UDP segment structure[2]</u>

```
<---16 bits -----><----16 bits------>
------------------------------------------
| source port      | destination port |
------------------------------------------
| length           | checksum         |
------------------------------------------
| payload                             |
------------------------------------------
```

Figure 2.2 UDP packet

<u>Key:</u>
Source port: port of the sending process
Destination port: port of the receiver process
Length: user datagram length
Checksum: provides error detection. Perform the 1's complement of the sum of the all 16 bits words in the segment. The receiver perform 1's complement of the received segment and add the checksum, if there are no errors the result must be 1's string.

2.1.2 IP protocol - network layer

The Internet Protocol (IP) is the most used protocol at the network layer, and it is responsible for routing datagrams between hosts and routers. In this work we used a IPv4 network which adds 20 bytes of overhead.

<u>IP Datagram format[3]</u>

32 bits

<-->

Version	Header	Type-of Service	Datagram length (bytes)
16-bit identifier		flags	13 bit fragmentation offset
time-to-live	upper-layer protocol	header checksum	
32-bit source IP address			
32-bit destination IP address			
options			
data if any			

Figure 2.3 IP Datagram

2.1.3 802.11b MAC, PLCP and PMD - Data link and Physical layer

The data link layer is used to move datagrams from one node to the next node on the route. Services provided at this layer depend on the upper layer protocol. The job of the physical layer is to transfer bits from one node to the other.

IEEE 802.11b standard splits the data link layer into Logic Link Control (LCC) and MAC sublayers, and physical layer is split in to the Physical Layer Convergence Procedure (PLCP) and Physical Medium Dependent (PMD) sublayers (figure 2.4). The standard specifies the functions of the MAC and both physical sub-layers, whilst the LCC sub layer is defined in the IEEE 802.2 standard [4] and the bridging between LAN’s is defined in IEEE 802.1 standard (figure 2.3).

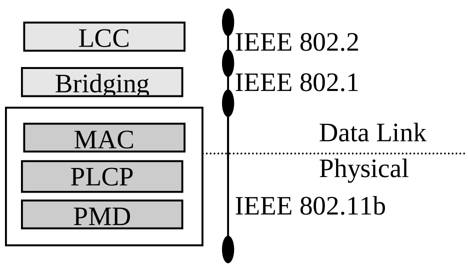

Figure 2.4 - 802.11b standard

2.2 Real-time protocols

2.2.1 RTP

The RTP standard[5] defines a packet structure for use in real-time applications, it includes fields for timestamps and other information that is useful in real time communications. RTP runs over UDP and data is encapsulated as it is shown in next figure.
The RTP protocol belongs to the application layer, but can be viewed as a sublayer of the transport layer. Data encapsulation is performed at the end systems and the protocol does not provide any mechanism to prevent out of order packets or other service quality features. In order to provide control functions, RTP is usually complemented with the RTCP protocol described in the next section.

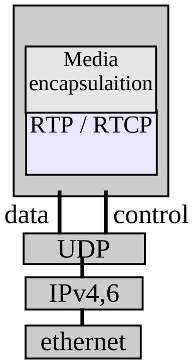

Figure 2.5

<u>Figure 2.6 shows the RTP packet format</u>

(24 bits header)

2	1	1	4	1	7	16
V=2	P	X	CC	M	PT	Sequence number
Timestamp						
Synchronization source (SSRC) identifier						
Contributing source (CSRC) identifiers						
Payload + paddings						

Figure 2.6 RTP Packet

Version (V=2): identifies the RTP version
Padding (P): if set indicates there are padding octets at the end of the packet
Extension (X): if set indicates the header is followed by a header extension
CSRC count (CC): contains the number of CSRC identifiers
Marker (M): used to mark relevant events
Payload type (PT): defines the format of the RTP payload
Sequence number: incremented by 1 each RTP packet
Time stamp: sampling instant of the first data octet contained in the payload
SSRC: identifies the synchronization source within the RTP session
CSRC: contributing sources for the payload (only 15 can be identified)

2.2.2 RTCP

The RTP control protocol (RTCP) is usually implemented in combination with the RTP protocol and it is based on the periodic transmission of control packets. RTCP doesn't send a payload with application data, instead it sends statistical information that is useful for the application program, providing feedback on the quality of the data and session control mechanisms. The way this information is used by the application layer is not defined in the RFC3550[5] standard and depends strictly on the application. RTCP and RTP packets are distinguished by using different port numbers (the RTCP port = RTP port +1). RTCP also runs over UDP.

RTCP packet types

Sender report(SR) and receiver report(RR): these reports provide feedback about reception quality. The only difference between them is that SR is used by active senders only and it has an extra field with sender information. If a node sent no data since the last report, then the application sends a RR, else it sends a SR.

Source description items(SDES): SDES is a tree level structured header and contains information describing source parameters, such as CNAME.

Indicates end of participation(BYE): indicates that a source is no longer active.

Application specific functions(APS): to be used by newly developed applications and features. It is an experimental header.

RTCP sender report format

2	1	5	7	16
V=2	P	RC	PT=RC=200	length
Synchronization source (SSRC) of sender				
NTP timestamp, most significant word				
NTP timestamp, least significant word				
RTP timestamp				
Sender's packet count				
Sender's octet count				
SSRC of first source (SSRC_1)				
Fraction lost			Cumulative number of packets lost	
Extended highest sequence number received				
Interarrival Jitter				
Last SR (LSR)				
Delay since last SR (DLSR)				
SSRC_2				
......				
Profile-specific extensions				

Figure 2.7 RTCP sender report

RTCP receiver report(RR) format

2	1	5	7	16
V=2	P	RC	PT=RC=200	length
Synchronization source (SSRC) identifier of sender				
SSRC of first source (SSRC_1)				
Fraction lost			Cumulative number of packets lost	
Extended highest sequence number received				
Interarrival Jitter				
Last SR (LSR)				

V=2	P	RC	PT=RC=200	length
Delay since last SR (DLSR)				
SSRC_2				
......				
Profile-specific extensions				

Figure 2.8 RTCP Receiver report

Chapter 3 Quality of service (QoS)

Quality of service(QoS) is an effort to manage transmission and error rates and to minimize latency, losses and jitter during network calls[13]. There are two classes of voice quality of service metrics: one is the objective metric which can be computed with high precision, and the other is the subjective metric, such as MOS, involving humans listening and assigning a rating to the audio quality.

3.1 IP metrics

Voice quality is related to three major factors: delay, losses, and jitter. Whilst data traffic is mostly affected by packet losses and resilient to delay, voice calls are loss tolerant but sensitive to delay and jitter (delay variance).

3.1.1 End-to-end delay

In voice communication “delay” usually refers to the end-to-end delay, that is the time a packet takes traveling from the sender process to the receiver process[VYDN1]. End-to-end delay has a significant effect on the perceived quality of IP telephony, and it can be the case that is it is not the same in both directions (asymmetric links). We assume that end-to-end delay is the sum of a processing delay, queuing delay, transmission delay and propagation delay[3].

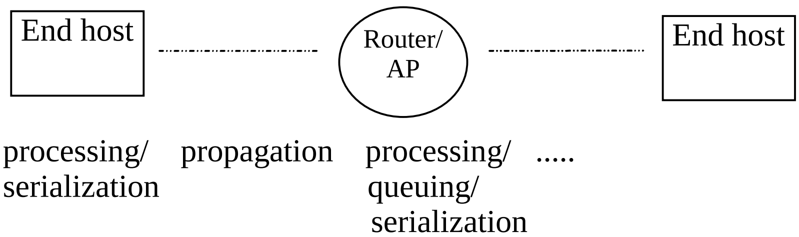

Figure 3.1 Delay contributions

(1) Packetization Delay
Time required by the nodes (access point, router, end host, etc.) to process the information and redirect the voice packets. The packetization delay is generally on the order of milliseconds.

(2) Queuing Delay
Delay experienced while the packet is waiting to be transmitted. It depends on both network traffic intensity, nature, and network design (link, equipment, structure ...). Typically in the order of microseconds to milliseconds.

(3) Serialization Delay
Time required to put all the packets onto the wireless link. Typically on the order of microseconds to milliseconds. The serialization delay can often be minimized thus increasing the link throughput.

(4) Propagation Delay
Time needed to propagate the information via the wireless and/or wired links that may exist between the sender and the receiver.

ITU Standard G.114 considers a phone calls quality as 'good' if the one-way delay is less than 150ms[9].

3.1.2 Jitter

Jitter is the packet inter-arrival statistical variation introduced by the IP network links (best effort network). Different definitions of jitter exist, however in this report we will adopt the IETF definition of jitter. Hence we define jitter as the mean deviation of the packet spacing at the receiver compared to the sender packet spacing for a pair of packets[10].

3.1.3 Packet losses

VoIP in 802.11b WLAN's runs over the UDP protocol which does not offer reliable delivery of the voice packets. During periods of congestion routers and/or access points may drop some packets in order to avoid overloading the system (or if the system is overloaded), generating losses. The ability our voice application has to deal with and tolerate packet losses depend, on the encoding and buffering techniques, some techniques allow as much as 50% loss rate.

3.2 Subjective metrics - Mean Opinion Scale (MOS)

MOS is a scale which rates the relative quality of voice conversations as perceived by human users. It is the most widely used subjective quality metric. Because the tests are taken by humans subjects it is useful to evaluate the network capability to support audio conversations, however its too expensive and time consuming for our use. Leading to the development of objective metrics.

As defined in the ITU-T Recommendation P.830[6] MOS defines perceived quality in a [0,5] scale as shown in the following table:

Rating	Description
5	Imperceptible errors
4	Perceptible but annoying
3	Sligthtly annoying
2	Annoying
1	Very Annoying

Table 3.1 MOS Rating

3.3 Objective metrics

MOS values can be calculated using objective metrics such as the ITU algorithm, Perceptual Evaluation of speech quality[7]. This metric offers advantages such as cost savings and is less time consuming. Others have used statistical methods to predict speech quality[8-10]. Implementing an objective metric is proposed as future work.

Chapter 4 Sphone and our real time tool

Sphone is a low delay VoIP communication tool[11]. The version we are using was developed at IMIT/KTH.

The aim of Sphone is to provide a platform for Internet telephony and it is distributed under the under the terms of the GNU General Public License as published by the Free Software Foundation.

Our extension was designed to provide Sphone users with real time QoS feedback. We calculate delay, jitter, and losses and we display these values on the screen at predefined intervals of time, so the user can have a clear perception of the link quality in real time.

Its is important to note that Sphone was designed for research purposes and the IP metrics are useful for this audience. These metrics also provide the basis to develop an objective link quality estimation for future use in both Sphone and other tools such as Minisip[12].

4.1 Architecture

Sphone was implemented in the ANSI C and it uses the RTP and RTCP real time protocols described in chapter 2. Sphone is based in two main modules: a sender and a receiver.
Because the sender and receiver are independent we chose to implement the real time tool in the receiver module for obvious reasons. First because we needed to know about the jitter and losses, and to compute them we had to use the receiver module to generate the arrival timestamps. Additionally because the missing information in the receiver was the RTT (wich can only be computed with the RR information), and we send it from the sender to the receiver using an RTCP extra field.

We also implemented a log file generator to allow offline statistical analysis, as described in section 4.3.

To keep the code modularized we added all our functions in a new file named “qos.c” under the directory “..sicsophone/src/qos.c” and the corresponding “qos.h” in “....sicsophone/include/qos.h”.

4.2 Computing delay, losses and jitter for our real time extension

We based on RFC3550[5] definitions of delay, jitter, and losses. The exact method we used is detailed in the following sections.

4.2.1 Delay

To display the delay in real time we added an extra field to the sender report, so we can have the RTT value on the receiver side.

Sphone uses the following fields from the RTCP headers to calculate the RTT delay:

<u>Last SR(LSR) – 32bits</u>: the middle 32 bits in the Network Time Protocol timestamps. It is perceived as part of the most recent RTCP sender report(SR) packet from source_n. If no SR has been received yet, the field is set to zero.

<u>Delay since last SR(DLSR) – 32 bits</u>: expressed in units of 1/65536 seconds, it represents the delay between the reception of the last SR packet from source_n and the moment when it is sent by this reception report block(RR).

Recording the arrival time of the RR allows the sender to estimate the delay using the LSR and DLSR fields from the receiver report (it could also be a SR if the receiver was also a sender, but this is not the case in Sphone).

RTT delay = [ARRIVAL] – LSR – DLSR

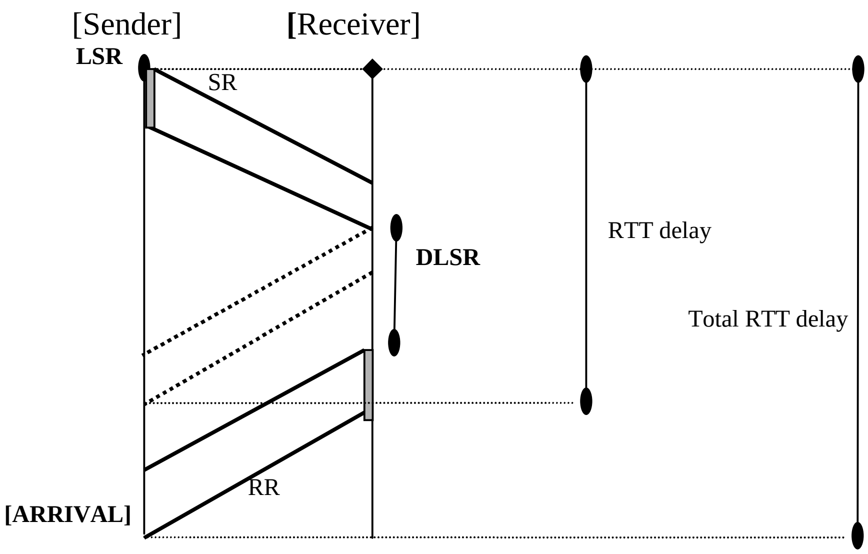

Figure 4.1 RTT timestamps

Note: See section 3.1.1 for delay definitions.

The RTT time is an approximation of the delay. We must note that the route is not symmetric and thus we can not assume that the one-way-delay is the half the RTT delay [9].

4.2.2 Jitter

To compute the jitter we use the time stamp field from the RTP header and a timestamps that we generate each time a RTP packet arrives. This information is stored in a dynamic list and at fixed intervals of time the jitter is computed for all stored packets using the recursive formula. Here Si is the RTP time stamp and Ri the time of arrival of the packet i (in RTP time stamp units, see figure 4.2). The inter arrival jitter is calculated recursively each time a packets arrives, according to the formula.

$$J_i = J_{i-1} + [\,|(R_{i-1}-R_i) - (S_{i-1},S_i)| - J\,] / 16$$

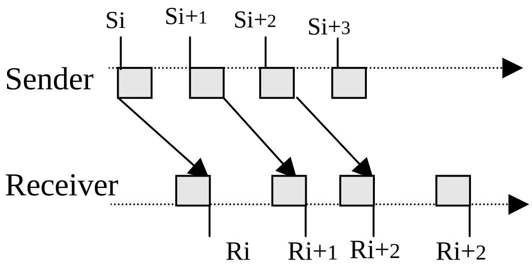

Figure 4.2 Inter arrival times

Jitter is reset to zero after each time interval. Packets that never arrive will not be included in the jitter calculation, we could include these packets as having infinite jitter, but this would disrupt our statistical results. Out of order packets are not used to compute jitter since this situation happens with a very low probability and it simplifies our algorithm.

4.2.3 Losses

We compute losses exactly as stated in RFC 3550. Analyzing the sequence number we calculate the number of packets lost as the the expected number of packets minus the number of packets received. The fraction of losses within a time interval is the number of lost packets divided by the number of expected packets. Every time this information is displayed these values are reinitialized to zero.

4.3 Real time displayed information

At regular pre-defined time intervals we display information regarding the number of disordered packet, number or repeated packets, number of packets and the three major IP metrics, namely: jitter, losses and delay (RTT). It is important to note that this values are computed for each time interval and then reinitialized to zero(as mentioned before) so we can have a real time perception of the channel quality.

An example of information displayed (here at 1 second intervals):

```
number of disordered packets:0; repeated packets:0; packets lost:0
number of packets in this list:48
percentage of losses:0.0 %
jitter is: 6,7 ms
last RTT is: 13 ms
--------------------------------------------------------------------------
number of disordered packets:0; repeated packets:0; packets lost:0
number of packets in this list:48
percentage of losses:0.0 %
jitter is: 7,0 ms
last RTT is: 13 ms
--------------------------------------------------------------------------
number of disordered packets:0; repeated packets:0; packets lost:0
number of packets in this list:48
percentage of losses:0.0 %
```

jitter is: 7,0 mS
last RTT is: 9 mS

4.4 Log files

When performing thousands of measurements it is important to store all the information so we can efficiently process it later. Another advantage of the traces is that we can use them as as input and reproduce experiments.

For each voice session Sphone creates a log file (*.log) in the "/sicsophone/traces" directory. The name of the file is "sicsophone" plus the GMT time and date in the format: gmt_yy_mm_dd-hh_mm_ss. We used GMT time because we can run simulations between different time zones and we may need to match the traces. An example of a log file name is: "sicsophone_gmt_2004-5-22_20-24-48.log".

<u>Trace format</u>:

A Sphone trace is basically a dump of the RTP header plus the packet arrival time in EPOCH[5] time format.

For example:

```
# SICSOPHONE LOG FILE TO USE IN POST PROCESSING
#Created on: sicsophone_gmt_2004-5-22_20-24-48.log
# E <- Data packet
# I  <- Source Changed and/or this is the first data packet from this source
# T <- first packet after a talk spurk
# D <- duplicated packet

#EPOCH:SEC uSEC   SZE        SSRC    SEQ  NTP:SEC FRAC
I  1085257488  46429  332   4157331870 606    46864      2635
E 1085257488 100255 332   4157331870 609    46864      6567
E 1085257488 120229 332   4157331870 610    46864      7877
RTT 13
-----------------------------------------------------------------------------
1       2                 3       4              5          6      7             8
```

Legend:
1 – Packet type
2 – Epoch time seconds (packet arrival time)
3 – Epoch time microseconds(packet arrival time)
4 – RTP packet size excluding header
5 – RTP source identifier
6 – RTP sequence number
7 – NTP timestamp seconds
8 – NTP timestamp fraction
RTT – Round Trip Time in ms

4.5 Validation of Sphone results

To test the Sphone time-stamping functionality we started a session using Ethereal and filtered the corresponding RTP packets so we could match the arrival timestamps from Ethereal with the timestamps we used in our real-time tool and recorded it in our traces. As an example, for packets with sequence numbers 1200, 1201 and 1202, over an ad-hoc link, the difference between Ethereal and Sphone timestamps were 60, 35 and 40 microseconds respectively. These differences are small and are probably due to the Sphone processing time that is not included in Ethereal. We can conclude that the timestamps are correct.

Using an external tool we could also process the log files and display the quality values each one second interval and checked that our real-time values calculations are correct. We did this using the number of packets in the list for each time interval.

Chapter 5 QoS measurements using our tool

The main goal of this work was to develop a real-time tool to display the quality of a voice call and create a recording of it. We performed measurements mostly to test and calibrate our tool, so these experiments are not extensive, but instead we focused on analyzing the results quantitatively. All the experiments were taken in the LCN/IMIT laboratory at the IT-University campus, Forum building, 8^{th} floor.

5.1 Ad-Hoc link

For this first experiment we setup an 802.11b wireless ad-hoc link between two laptops runing Fedora Core 1 Linux and wireless cards.

In a command shell (as root) we typed :

```
#  iwconfig eth1 essid "sphone"
#  iwconfig eth1 mode ad-hoc
#  ifconfig eth1 ip 10.0.0.1 | 10.0.0.2 for the other computer
```

We started our test with 2 meters distance between the laptops and then started to move one of the mobile nodes arround. The results of our simulation, taken at intervals of one second are shown in the following table:

Time(sec)	Distance/location	RTT(ms)	Jitter(ms)	Packets lost	Fraction lost	Repeated packets	Disordered packets
1	From 2 to six meters	-	0	0	0	0	0
2		-	0.7	0	0	0	0
3		-	1.2	0	0	0	0
4		-	0	0	0	0	0
5		-	0	0	0	0	0
6		1	0.7	0	0	0	0
7		1	0	0	0	0	0
8		1	0.8	0	0	0	0
9		1	2	0	0	0	0
10		5	4.6	0	0	0	0
11	Going away and	5	4.6	0	0	0	0
12		5	2.4	0	0	0	0
13		5	3.8	0	0	0	0
14		5	2.7	0	0	0	0
15		15	2	0	0	0	0
16		15	3.5	0	0	0	0
17		15	3.9	0	0	0	0
18		15	3.2	0	0	0	0
19		12	7	0	0	0	0
20		12	7.9	0	0	0	0
21		12	6.9	0	0	0	0
22		6	6.6	0	0	0	0

Time(sec)	Distance/location	RTT(ms)	Jitter(ms)	Packets lost	Fraction lost	Repeated packets	Disordered packets
23	Moving inside some other office	6	6	0	0	0	0
2		6	0.9	0	0	0	0
25		3	1.9	6	12.5	0	0
26		3	0.9	9	18.83	0	0
27		3	2	5	10.2	0	0
28		5	2	9	18.83	0	0
29		5	2	21	63.6	0	0
30		5	5	17	34.7	0	0
31		14	7	59	78.3	0	0
32		14	6	4	87.9	0	0
Average values							
-	**-**	**7.5**	**3.04**	**-**	**10.00%**	**-**	**-**
Totals							
-	**-**	**-**	**-**	**130**	**-**	**0**	**0**

Table 5.1 802.11b Ad-hoc measurements

As expected when both computers were close to each other we have no losses and both the RTT and jitter are very low. When the distance increases we start experiencing higher jitter and RTT, but the losses remain low until one of the nodes moves behind some obstacles. At this point we detected losses, greater jitter and RTT. Finally one of the nodes moved into another room and we experienced a high percentage of losses as we expected. Since Sphone codecs are not loss tolerant we experience degraded voice quality when facing losses. However this is not the case of most codecs, wich can tolerate the percentage of losses we experienced in most cases.

Note: each value of RTT is repeated several times because the sender only sends a sender report (which has as an extension with the last RTT computed) every 4 seconds. At the start there is no RTT displayed because the receiver program had not received any sender report so far.

5.2 Experiment using LCN wireless Network

In this experiment we connected two mobile nodes to the KTH “open” wireless network and generated voice sessions with each computer in a different office. We will not present the evolution of losses, jitter and RTT, instead we present only the average values of our parameters. The average results for a series of some voice sessions (each 70 seconds), taken on 24/05/2004, starting at 17:37 are:

RTT(mS)	Jitter(mS)	Fraction lost	Repeated packets	Disordered packets
12	3.7	4.55%	0	0

Table 5.2 “Open” Voice-over-WLAN measurements

The jitter and losses figures are very low, 3.7 ms and 4.55% respectively. And the average delay is approximately 6 ms which is well below the ITU-T 150 ms limit. We considered the one-way delay as RTT/2 since we are assuming the link is symmetric, which is a good approximation for our simulation within LCN lab, but may not be for long distance calls.

Chapter 6 Conclusions and Future work

We concluded that voice over IP is well supported by the KTH's "open" wireless network if both nodes are inside the LCN lab, however, more experiments including different times of the day would be needed to support this conclusion.

Our decision of not counting jitter for out of order packets proved correct since we did not get any disorded packets in our tests.

Our real-time measurement tool proved to be useful to adjust and calibrate experiments as it provides a useful QoS feedback. Together with the log files we expect that our tools will be helpful and represent an improvement to Sphone.

Our add-on currently only supports the VoIP tool Sphone. In the future we would like to extend its functionality to other VoIP applications. There are some other improvements that we could not implement due to lack of time. For instance, the quality value that our tool gives is updated every second, but we would like to add some flexibility to be able to select the update frequency before or even during the voice session. Also, a graphical GUI with a bar that shows the quality between a minimum and a maximum would be more user-friendly than just pure text-only information, as our tool shows now (this could be achieved implementing an objective metric algorithm as mention on page 10).
An application to process the log files in under development but was not ready in time for this course, however it will be presented as part of my thesis work.

References

[1] IEEE 802.11b IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. Institute of Electrical and Electronics Engineers, November 1999 Edition.

[2] IETF, Request For Comments:768, RTP: A Transport Protocol for Real-Time Applications, 28 August 2003 edition.

[3] James F. Kurose, and h W. Ross, Computer Networking - a top down approach featuring the internet, International Edition, 2003

[4] IEEE 802.2 Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks.Part 2: Logical Link Control, 1998 Edition.

[5] IETF, Request For Comments: 3550, RTP: A Transport Protocol for Real-Time Applications, July 2003 edition.

[6] International Telecommunications Union, Subjective Performance Assessment of Telephone-band and Wideband Digital Codecs ITU-T Recommendation P.830, 1996

[7] ITU-T Recommendation P.862, Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs.

[8] L.A.R. Yamamoto, and J.G.Berends, Impact of network performance parameters on the end-to-end perceived speech quality, Expert ATM Traffic Symposium, Mykonos, Greece, Sep. 1997.

[9] S. Mohamed, F. Cervantes-Perez, and H. Afifi, Integrating Networks Measurements and Speech Quality Subjective Scores for Control Purposes, IEEE Infocom 2001.

[10] S. Mohamed, F. Cervantes-Perez, and H. Afifi, Audio Quality Assessment in Packet Switched Networks: an “Inter-Subjective” Neural Network Model, Proc. International Conference on Information Networks, Japan, Jan. 2001.

[11] Olof Hagsand, Ian Marsh and Kjell Hanson, Sweden, “Sicsophone: A low delay internet telephony tool”, 29th Euromicro, September 2003.

[12] J. Bilien, ***Key Agreement for secure Voice over IP***, Master Thesis, KTH, Stockholm, Sweden, December 2003

[13] Luan Dang, Cullen Jennings, and David Kelly, *Practical VoIP: Using VOCAL*, O'Reilly, 2002, ISBN 0-596-00078-2.