Quality and Performance Evaluation of VoIP End-points

Wenyu Jiang, Kazuumi Koguchi and Henning Schulzrinne
Department of Computer Science, Columbia University
{wenyu,koguchi,hgs}@cs.columbia.edu

ABSTRACT

We evaluate the quality and performance of a number of hardware and software Voice over IP (VoIP) end-points. In particular, we focus on the following aspects: mouth-to-ear (M2E) delay, clock skew, silence suppression behavior and robustness to packet loss. Our preliminary results show that mouth-to-ear delay depends mostly on the receiving end-point, and hardware IP phones, when acting as receivers, usually achieve a low average M2E delay (45-90 ms) under low jitter conditions. For software VoIP clients (as receivers), their average M2E delays range from 65 ms to over 400 ms, depending on the actual software. All the VoIP end-points we tested are able to compensate for clock skew, although some exhibit the symptom of occasional playout buffer underflow. Only a few of the tested end-points support silence suppression. We find that their silence detectors have a fairly long hangover time ($>$ 1 sec), and they may falsely detect music as silence. When packet losses occur, we find that all tested hardware IP phones support some form of packet loss concealment better than silence substitution. The concealment generally works well for up to two consecutive losses at 20 ms packet intervals, but bails out (voice fading quickly) after the third consecutive loss.

1. INTRODUCTION

Voice over IP (VoIP) is a service that requires synergy between the underlying network for transport and the end-points responsible for voice processing. There are extensive literature on network quality of service (QoS), such as in the areas of DiffServ [1] and IntServ [2]. In contrast, little has been done on studying the QoS of VoIP end-points. Depending on the implementation, such as what playout algorithms [7, 9] these end-points use, whether they are hardware or software based, their performance may differ dramatically.

In this paper we evaluate the QoS of a number of VoIP end-points, including several well-known brands of IP phones and some popular software VoIP clients. The QoS aspects we examine include mouth-to-ear (M2E) delay, clock skew, silence suppression behavior and robustness to packet loss. While most studies of QoS focus on bounding the network delay, we examine the M2E delay because that is what matters to the end-users. In particular, we shall see in later sections that even in a LAN environment where delay is virtually zero, the M2E delay can be quite high for some VoIP implementations. Our preliminary evaluation reveals the following interesting results:

- Firstly, the M2E delay depends mainly on the receiver [1]. Most hardware IP phones achieve a low M2E delay, typically between 45 and 90 ms [2]. By contrast, the situation is a lot more diverse for software VoIP clients. Microsoft Messenger has the lowest M2E delay of 65-120 ms, while NetMeeting has the highest M2E delay of over 400 ms. Even more surprisingly, while Messenger XP has been touted as the best Messenger implementation, our experiments show that the Windows 2000 version of Messenger achieves a consistently lower M2E delay, with exactly the same hardware.
- Secondly, clock skew is present between most end-points, but we find that all of them are able to compensate for it with adaptive playout buffering, albeit some of them seem to suffer from playout buffer underflow occasionally.
- Thirdly, we find that only a few of our tested end-points support silence suppression, but luckily all end-points are able to adjust playout delay adaptively even when the sender transmits continuously. The silence detectors they use have long hangover time ($>$ 1 sec). However, one of the detection algorithms may falsely treat music as silence, creating a gap in the output audio.
- Finally, when encountering packet losses, all the hardware IP phones support some form of packet loss concealment (PLC) better than silence substitution. Their PLC generally works well for up to two consecutive losses at 20 ms packet intervals, but the voice starts to fade quickly after the third consecutive loss. This is a reasonable behavior because PLC without fading will result in some annoying buzzing or repetitive sound.

The remainder of this paper is organized as follows. Section 2 lists related work. Section 3 describes the setup of our experiments. Section 4 presents the measurement results, and Section 5 and 6 concludes the paper and lists future work.

[1]Hence whenever we refer to an end-point's M2E delay, we imply that it is acting as a receiver

[2]By default we always assume a LAN test environment

2. RELATED WORK

There have been many studies [1, 2] on limiting the network delay to facilitate real-time multimedia applications. However, little has been done on studying the actual end-to-end (mouth-to-ear) delay for these applications or hardware implementations.

Sage Instrument has a few products such as 935AT (http://www.sageinst.com/WireNavTemplateProducts935AT.htm) for measuring round-trip delay and detecting packet loss. The 935AT allows one-way delay measurement however, only for analog 2-wire circuits, and not for 4-wire circuits or IP phones.

3. EXPERIMENT SETUP

3.1 Measurement Approach

Our approach to measure mouth-to-ear delay is by recording both the original audio feed and the receiver's output audio in a two-channel (stereo) mode, as illustrated in Figure 1. The couplers in Figure 1 are special devices that connect between the handset and the base of a telephone (including an IP phone). They allow either feeding audio into a telephone or receiving audio from a telephone, although not simultaneously.

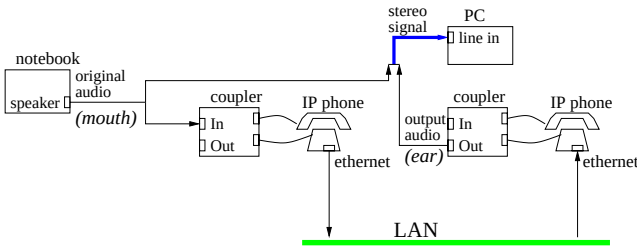


Figure 1: Measuring mouth-to-ear (M2E) delay

To automate delay estimation, we use the adelay software from our lab. It reads a stereo audio file and calculates a most likely delay offset based on auto-correlation in the frequency domain. We have tested its correctness by inserting a known delay offset with a special audio mixer. The precision of adelay is always within 1 ms in these tests.

For VoIP to be successful, first of all it has to work well at least in an ideal environment, and that means achieving a low M2E delay, free from playout buffer underflow/overflow, etc. Therefore, except noted, all of our experiments are carried out in a LAN environment.

The audio source in our experiments are digital recordings of an audio book tape cassette.

3.2 End-point Devices and Configurations

We evaluate IP phones from three major vendors: Cisco, 3Com and Pingtel, plus a 1-line PSTN[3]/IP gateway by Mediatrix, which may be seen as a form of hardware IP phone. The software VoIP clients include Microsoft Messenger and NetMeeting (both running on a AMD K7 PC with Windows 2000/XP dual-boot, and a Pentium-3 notebook with Windows XP), sipc [6], and Net2Phone. Finally, we tested the delay of mobile (GSM) phone to PSTN for reference. Table 1 lists all tested end-points and whether silence suppression is used. sipc uses the Robust Audio Tool (rat) [4] as the underlying multimedia engine. rat supports silence suppression, but its default is to not use it. The codec that Net2Phone uses may be G.723.1, but we cannot be sure because Net2Phone does not use RTP [11] encapsulation and hence we cannot determine the media payload type.

[3]The Public Switched Telephone Network in our daily life

End-point	platform/model	silence suppression
Cisco phone	7960	Y
3Com phone	NBX	N
Pingtel phone	Xpressa	N
Mediatrix gateway	APA III 1FXS	N
sipc	Solaris, Ultra-10	N
Messenger	Win 2000/XP, K-7 & P-3	Y
NetMeeting	Win 2000/XP, K-7 & P-3	Y
Net2Phone	Win NT, P-2	Y
GSM phone	GSM 1900 US	n/a

Table 1: List of tested end-points

Table 2 shows some of the configurations in the experiments. All end-points implement the G.711 μ-law [5] codec. However, not many other codecs are supported. Due to limitations in interoperability and time constraint, we cannot test all pair combinations of end-points.

parameter	case	value
codec	default	G.711 μ-law (64 kb/s)
	Cisco phone	G.729 (8 kb/s)
	NetMeeting	G.723.1 (6.3 kb/s)
	Net2Phone	G.723.1?
packet interval	default	20 ms
	Mediatrix GW	30 ms
	NetMeeting	30 ms
	Net2Phone	60 ms

Table 2: Common parameters in the experiments

4. MEASUREMENT RESULTS

4.1 Mouth-to-ear (M2E) Delay without jitter

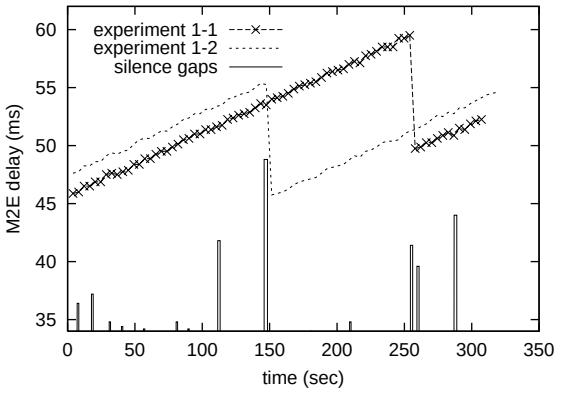
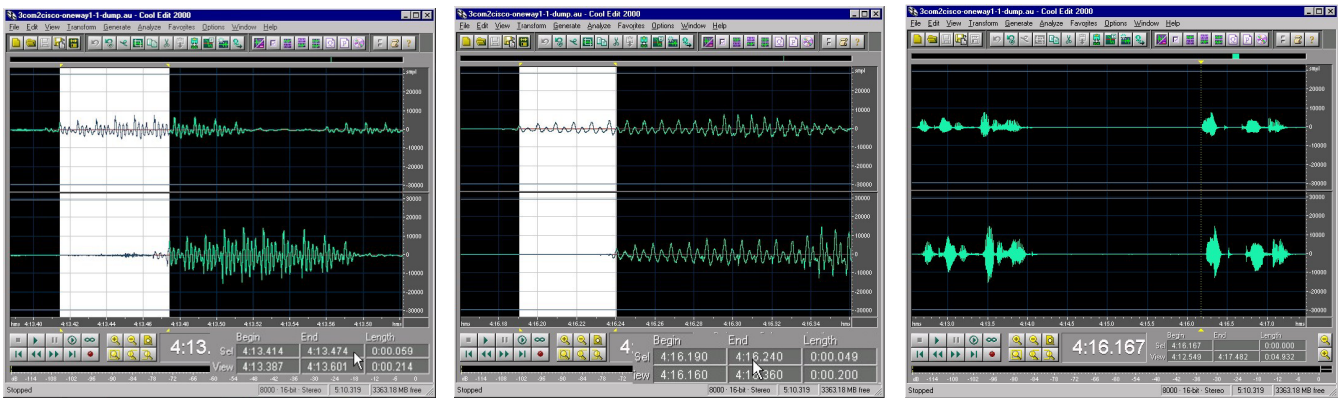


Figure 2: 3Com to Cisco, gaps *not* drawn to scale

Figure 2 shows an example plot of M2E delay and those silence gaps that are longer than 1 sec. The notation "experiment 1-2" means part 2 of call no. 1. Splitting audio into parts results in smaller files and allows faster interactions between the experimenters and the processing of data. Between part 1 and part 2 is a short pause when we save part



(a) M2E delay reaching a peak value of 59 ms (b) M2E delay drops by 10 ms to 49 ms right after the peak (c) The time period (and silence gap) during which the M2E delay dropped

Figure 3: Visual confirmation of M2E delay decrease in Figure 2, experiment 1-1; left channel is the original audio (mouth), right channel is the output audio (ear)

1's audio to disk. In experiment 1-1, the M2E delay from 3Com to Cisco reaches a peak of about 59 ms at the time of 254 seconds, then drops to 49 ms immediately. The highly linear trend is due to clock skew. A similar linear trend and drop in M2E delay are in experiment 1-2. The drop is also 10 ms. In fact, we find that the Cisco and Pingtel phone always adjust delay by 10 ms granularity, but due to limited space, we cannot show all delay plots. The 3Com phone and the Mediatrix gateway have 5 ms granularity.

The delay drop for experiment 1-1 in Figure 2 is visually confirmed in Figure 3. Figure 3(c) shows a long silence gap during the drop. This concurs with the convention that playout delay should only be adjusted at the beginning of a new talk-spurt [7, 9], an event represented by an RTP [11] packet with the marker (M) bit set to 1. However, according to Table 1, the 3Com phone does not use silence suppression. Hence its RTP packets will never have the M-bit set to 1. But the Cisco phone is still able to adapt the playout delay irrespective of the M-bit, although the point of delay adjustment still occurs right after a silence gap.

	3Com	Cisco	Mediatrix	Pingtel	sipc
3Com (range)		51.4 ms (2.5 ms)	47.1 ms (3.1 ms)	55.1 ms (2.9 ms)	223 ms (8.7 ms)
Cisco	63.0 ms (3.8 ms)	75.8 ms	78.1 ms (8.4 ms)	74.0 ms (8.6 ms)	206 ms (12.3 ms)
Mediatrix	85.8 ms (16.3 ms)	77.6 ms (4.0 ms)		72.5 ms (10.7 ms)	
Pingtel	46.6 ms (4.1 ms)	63.0 ms (8 ms)	57.6 ms (1.8 ms)		230 ms (63.4 ms)
sipc	52.4 ms (3.1 ms)	59.8 ms (0.9 ms)		64.2 ms (16.4 ms)	
Cisco G.729		98.7 ms			

Table 3: Average M2E delays, IP phones and sipc

Instead of showing all test pairs' delay plots, Table 3 summarizes them with the average M2E delay. Its first column representing the sender, and its first row denoting the receiver. The numbers in the parentheses are the difference (range) between the highest and lowest average M2E delay among all calls for the same pair of end-points. A low range indicates high repeatability, and this is clearly true for Table 3 in most cases. All IP phones (including the Mediatrix gateway) achieve a delay below 90 ms, and in most cases below 65 ms. Pingtel to 3Com has the lowest average delay of 46.6 ms. Between two Cisco phones, delay under G.729 (98.7 ms) is nearly 15 ms higher than under G.711 (75.8 ms), clearly due to additional compression delay for G.729.

Between the IP phones, due to the low delay, it is not very clear whether the sender or the receiver plays a more dominant role in M2E delay. However, the role becomes evident when sipc is the receiver. Its M2E delay is consistently above 200 ms irrespective of the sender.

The dominant role of the receiver is more evident in Table 4. For example, Messenger XP (notebook) to XP (pc) achieves an average delay of 120 ms. However, when the receiver switches to Messenger for Windows 2000 (same pc), the delay consistently drops to 68.5 ms, indicated by the small range in Table 4. This again confirms that the receiver dominates the M2E delay. It is also surprising because Messenger XP has always been claimed as the best Messenger implementation, while we find that Messenger 2000 actually has lower delay, with exactly the same hardware.

end-point A	end-point B	A → B	B ← A
MgrXP (pc) (range)	MgrXP (notebook)	109 ms (0.8 ms)	120 ms (44.6 ms)
Mgr2K (pc)	MgrXP (notebook)	96.8 ms (5 ms)	68.5 ms (10.1 ms)
NM2K (pc)	NMXP (notebook)	401 ms (46.9 ms)	421 ms (6.8 ms)
Net2Phone	PSTN	288 ms (77.3 ms)	371 ms (12.2 ms)
Mobile (GSM)	PSTN	115 ms (4.8 ms)	109 ms (5.1 ms)

Table 4: Average M2E delays for PC clients and mobile phone

The sender does in some cases play a minor role in M2E delay. When Messenger XP (laptop) acts as the receiver, the average delay is 109 ms with Messenger XP (pc) as sender, but drops to 96.8 ms when sender is Messenger 2000. It could be that Messenger 2000 manages to reduce the processing time required before sending out every packet.

The predecessor of Messenger is NetMeeting, and its M2E delay is a very high 400 ms, using the same PC and note-

book. This is quantitative evidence that NetMeeting is not well designed for real-time, interactive conversations.

The last VoIP end-point we test is the Net2Phone CommCenter, a quite popular application in US. Its M2E delay is unfortunately, also quite high, near or above 300 ms. We deliberately measured the round-trip time (RTT) between the PC running Net2Phone and the PSTN gateway it uses. The RTT usually ranges between 5-30 ms. Therefore the 300 ms M2E delay is not caused by the network, but by the client or the gateway, and most likely at the receiving end.

Finally, we called a local phone number from a GSM mobile phone. The resulting delay in either direction is close to 110 ms. Comparing with Messenger XP, we can say that Messenger achieves a delay similar to that of a mobile phone.

4.2 Clock Skew

Clock skew can cause playout buffer underflow or overflow after a certain period of time, resulting in occasional choppy audio. In our experiments, all the end-points are able to adjust the playout delay whenever clock skew causes the M2E delay to go too high or too low, such as in Figure 2, and irrespective whether the sender supports silence suppression. For the 3Com and Pingtel phone, they appear to suffer from playout buffer underflow even when there is no packet loss or delay jitter. Because such events usually occur when the delay is being adjusted, we think they are caused by the playout algorithm.

Next we analyze the rate of clock skew. Our notation is: if the clock drift rate from A to B is r, then a positive r means the M2E delay has an increasing trend, thus A's clock is faster than B's. Ideally clock drift is symmetric, that is, the drift rate from B to A should be $-r$. This is well observed in Table 5.

	3Com	Cisco	Mediatrix	Pingtel	sipc
3Com		55.4	43.3	41.2	-333
Cisco	-55.2	-0.4	-11.8	-12.1	-381
Mediatrix	-43.1	11.7		-0.8	
Pingtel	-40.9	12.7	2.8		-380
sipc	343	403		376	
Cisco G.729		-0.4			

Table 5: Average clock drift rates (in ppm) for IP phones and sipc

The drift rate depends on the crystal oscillator used to drive the voice circuitry, but its magnitude usually falls within 100 ppm (parts per million), with 25 ppm being typical [12]. Between the IP phones (including the Mediatrix gateway), the magnitude of the combined drift is within 60 ppm. However, for sipc it is always higher than 300 ppm, far beyond 100 ppm. Between two Cisco phones, there is almost no clock skew, presumably because they use the same type (or maybe even the same batch) of oscillators.

end-point A	end-point B	A → B	B ← A
MgrXP (pc)	MgrXP (notebook)	172	87.7
Mgr2K (pc)	MgrXP (notebook)	165	85.6
NM2K (pc)	NMXP (notebook)	?	-33
Net2Phone	PSTN	290	-287
Mobile (GSM)	PSTN	0	0

Table 6: Average clock drift rates (in ppm) for PC clients and mobile phone

Table 6 shows the drift rates for Messenger and other PC clients. Surprisingly, for some unknown reason, Messenger has a positive drift rate in either direction, but interestingly, the drift in the A → B direction is about twice that of B←A. NetMeeting exhibits highly oscillative delays, as a result its values are not reliable. The drift rates of Net2Phone are very symmetric, and relatively high, nearly 300 ppm and quite close to that of sipc. Since Net2Phone runs on a Pentium-2 NT PC, high drift rates may indeed be typical on a PC or workstation (sipc). Finally, the mobile phone exhibits no sign of drift, because the mobile phone network and the PSTN is known to share a common, global stratum-1 clock.

4.3 Behavior under Packet Loss

When a packet is lost in the network, if no retransmission or forward error correction [10] is used, the receiver is responsible for repairing the lost packet with some approximation, a procedure known as packet loss concealment (PLC) [8]. The simplest method is silence substitution, by replacing the lost waveform with silence. It however, produces the worst quality. A better option is packet repetition, by repeating waveform in the last packet. A further improvement is extrapolation (instead of repetition) of last packet. Finally, the best method is interpolation, if the packets before and after the loss(es) are received in time. All these methods usually fade the voice gradually to prevent any annoyingly repetitive or buzzing sound.

We evaluated packet loss behaviors on the IP phones (Cisco, 3Com, Pingtel). Packet loss is simulated by a UDP relay program that connects from sipc to an IP phone. For easy verifiability, the relay program generates deterministic bursty packet losses. For example, it can generate 5 consecutive losses for every 100 packets received (denoted as 5/100). We tested 1/100, 3/100, 5/100, 10/100 and 1/20 for all three phones. Our result shows that: first, all of them implement a PLC better than silence substitution. Second, their PLC usually works for up to two (three for Cisco) consecutive losses at a 20 ms interval, then the voice immediately fades to silence. This is acceptable because repairing more than three consecutive losses becomes extremely difficult and it may cause side effects like buzzing sound. Thirdly, PLC does not increase the M2E delay in any affirmative way.

IP phone	PLC used	loss tolerance	loss pattern 3/100	loss pattern 1/20
Cisco	extrapolation or interpolation	3 packets	almost inaudible	almost inaudible
3Com	extrapolation only?	2 packets	audible	very audible
Pingtel	packet repetition with edge smoothing	2 packets	audible	very audible

Table 7: PLC behaviors of IP phones, G.711 μ-law, 20 ms interval

By examining the output waveform, we find that the Pingtel phone repeats the waveform, but it does some smoothing because there are no sharp, discontinuous edges between frames. Both Cisco and 3Com phones use at least extrapolation, but it is very hard to distinguish whether they use interpolation at all just by examining the waveform. Theoretically interpolation can be done if the loss gap is shorter than the playout delay, but we would not know whether the Cisco and 3Com phone opt to do so. Table 7 summarizes the behaviors of these IP phones. It also shows the relative

quality of PLC among the phones, that is, how audible is the audio distortion. The quality rating is given by the authors by listening to the output audio, but the difference between the three levels (almost inaudible, audible and very audible) are very clear-cut. The Cisco phone is the most robust. In particular, for the 1/20 loss pattern (effectively 5% loss), its quality is still very good. By contrast the 3Com phone performs noticeably worse in the 1/20 case. With only one consecutive packet loss, the loss gap (20 ms) is very likely to be less than the playout delay and would thus allow interpolation. However, the fact that the 3Com phone has a much worse quality suggests it is not doing interpolation even for 1/20 case. Therefore in Table 7 we tentatively mark it as extrapolation only. The Pingtel phone performs similarly as 3Com, although it uses packet repetition, so occasionally one can hear some repetitive or buzzing sound. Finally, although not shown here, when the loss pattern is 1/100, there is no audible distortion for any of these IP phones.

Figure 4 shows how the M2E delay of the Pingtel phone changes over time under different loss patterns. Clearly PLC does not introduce any extra delay in any affirmative way. The Cisco and 3Com phones have similar properties.

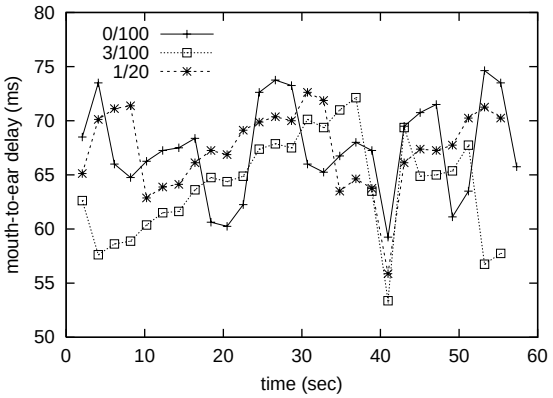


Figure 4: Evolution of M2E delay under packet loss, sipc to Pingtel

In brief, the packet loss concealment behaviors of all three IP phones are acceptable, and the Cisco phone exhibits the highest level of robustness.

4.4 Silence Suppression Behavior

Silence suppression serves several purposes. First, it saves bandwidth by transmitting only when talking. Second, it may save processing power at the receiver and possibly at the sender too. A nice example is tele-conferencing. Without silence suppression, the conferencing mixer has to decode and mix voice packets from all participants. This can consume a lot of computing power for a large size conference. With silence suppression, however, only the voices of active speakers need to be mixed. This greatly lowers the mixer's processor requirement and reduces cost. Third, silence suppression facilitates per talk-spurt adaptive playout delay adjustment.

4.4.1 Hangover Time

The hangover time is the duration by which a silence detector delays its final decision. Its main purpose is to avoid clipping the end of a speech segment. When a fixed hangover time is used, a value of 200 ms is usually sufficient [3].

In our experiments the end-points' hangover times were not specified to us. Therefore we measured their values using waveforms consisting of several ON (sine wave) and OFF (silence) periods. The OFF periods are 10 seconds long to guard against long hangover time. Our results are summarized in Table 8. The Cisco phone has a fairly long hangover time. NetMeeting's hangover time is the shortest, but still longer than the 200 ms used in [3]. The Cisco phone generates comfort noise packets as well.

Sender	Hangover time	Comfort noise packets?
Cisco	2.3-2.36 sec	Y
Messenger	1.06-1.08 sec	N
NetMeeting	0.56-0.58 sec	N

Table 8: Silence detector properties

There is nothing quite wrong with a long hangover time. It is safer and prevents end-clipping of speech, but it does use slightly more bandwidth.

4.4.2 Robustness for Non-speech Signals (Music)

Although silence detectors are usually designed to detect speech activity only, it is desirable if they work with non-speech signals as well. For example, when playing on-hold music to a waiting customer, it is essential that the music be played without any jerkiness or unnatural gaps.

We find that none of the three silence detectors in Table 8 works perfectly for music input. They may falsely predict music as silence, leading to annoying, unnatural gaps in the output audio. The Cisco phone is relatively more robust in dealing with music, but it still fails sometimes.

sender	input level	no. of gaps	total gap length
Cisco phone	-23.1 dB	1	1 sec
	-25.3 dB	1	2 sec
	-31.4 dB	1	3 sec
	-36.7 dB	2	5 sec
	-41.4 dB	14	10 sec
	-43.5 dB	18	25 sec
Messenger 2000	-19.5 dB	2	5.5 sec
	-23.9 dB	13	12 sec
	-24.5 dB	19	14.5 sec
	-25.4 dB	29	18 sec

Table 9: Silence detector performance for music (2.5 minutes long)

Table 9 shows the total number and duration of unnatural gaps created by the Cisco phone and Messenger 2000, at various audio input levels. Messenger clearly uses a higher threshold than the Cisco phone for silence detection. The standard speech level is -26 dB, but a level of -36 dB should also be acceptable. In fact, we also tried the input level close to -43.5 dB on the Cisco phone for speech, and we did not notice any unnatural gaps. This indicates that even if the silence detector works for speech, it may not for music.

4.5 M2E Delay under Network Jitter

Finally, we have performed an initial test on how jitter affects the M2E delay. We used the same UDP relay program used for packet loss tests, and added code to insert delay based on a packet trace. We used one of the typical packet

traces collected between a university machine and a PC with cable modem. It is well known that cable modem uplink exhibits high jitter. The 99% delay is 43.8 ms for the downlink trace we used, and 93.6 ms for the uplink trace, nearly 50 ms higher. Figure 5 illustrates the experiment results with the Cisco phone as the receiver. The average delay is 38.4 ms higher in the uplink case. This increase is slightly less than the 50 ms difference in the 99% delay between the two traces, which is good delay-wise. In addition, we do not notice any distortion in the output audio. Therefore, the Cisco phone does a good job of playout buffering and results in almost no late loss, if any.

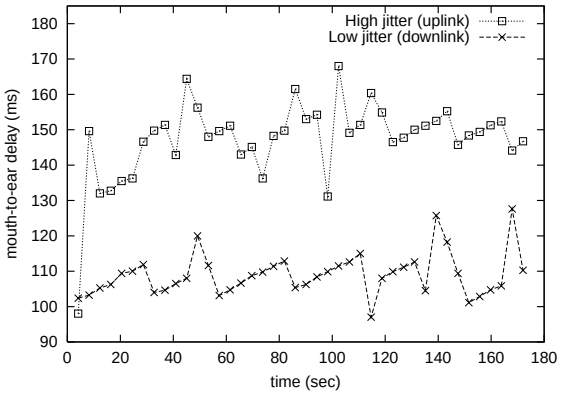


Figure 5: Impact of jitter on M2E delay, sipc to Cisco, typical cable modem delay trace

5. CONCLUSIONS

We have performed a wide range of measurements evaluating various VoIP end-points. We find that the mouth-to-ear (M2E) delay depends more on the receiver. Most hardware IP phones can achieve a low M2E delay, typically below 65 ms on average. The software-based clients, however, have diverse results. Among them, Messenger 2000 has the lowest M2E delay, only 68.5 ms on average. This is even better than Messenger XP, which achieves an average of 97-120 ms. sipc achieves 200-230 ms in comparison. NetMeeting performs the worst, with over 400 ms on average. Net2Phone also has a high M2E delay (290-370 ms). As a reference, a GSM mobile phone incurs about 110 ms of delay. Therefore, Messenger 2000 is even better delay-wise than GSM. The low delay of Messenger 2000 compared to other software clients (especially NetMeeting on the same hardware and OS) suggests that the application is a more important determinant of delay than the OS or device driver.

All the end-points we tested can compensate for clock skew, although some appear to have playout buffer underflow occasionally. The drift rates are quite high (near or more than 300 ppm) for sipc (running on a workstation) and Net2Phone (running on a PC).

Silence detectors are enabled in Cisco phone, Messenger, NetMeeting, and Net2Phone. The first three have long hangover times, ranging from 0.56 sec (NetMeeting) to 2.36 sec (Cisco). However, they do not work well for music input, leading to unwanted gaps in output audio.

All these results are based on a LAN environment, with virtually no loss, delay and jitter, serving as a baseline reference. When operating over the Internet, packet losses may occur. Therefore we evaluated the quality of three IP phones (Cisco, 3Com, Pingtel) under packet loss. All of them implement acceptable packet loss concealment (PLC) algorithms. The Cisco phone is the most robust. Our results also indicate that PLC does not cause any increase in M2E delay.

Transmission over the Internet also introduces more delay. The fixed portion of network delay adds to the M2E delay under the LAN environment directly as a constant. The variable portion of network delay (jitter) has a more complex effect depending on the playout algorithm being used. Our preliminary test on jitter shows that the Cisco phone is able to tolerate typical cable modem uplink jitter at a moderate increase in M2E delay and with no audible distortion due to late loss. Furthermore, when operating the Internet.

6. FUTURE WORK

There are several aspects we plan to investigate further. For example, measuring the sensitivity (threshold) of silence detectors; analyzing jitter behavior on more IP phones and software clients; repeating the Net2Phone experiment on the same PC that runs Messenger 2000, to check whether the machine/OS makes a big difference in delay and clock skew.

7. REFERENCES

[1] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and Weiss, W. An architecture for differentiated service. RFC 2475, Internet Engineering Task Force, Dec. 1998.

[2] Braden, R., Zhang, L., Berson, S., Herzog, S., and Jamin, S. Resource ReSerVation protocol (RSVP) – version 1 functional specification. RFC 2205, Internet Engineering Task Force, Sept. 1997.

[3] Brady, P. T. A statistical analysis of on-off patterns in 16 conversations. *Bell System Technical Journal 47*, 1 (Jan. 1968), 73–91.

[4] Hardman, V., Sasse, A., Handley, M., and Watson, A. Reliable audio for use over the Internet. In *Proc. of INET'95* (Honolulu, Hawaii, June 1995).

[5] International Telecommunication Union. Pulse code modulation (PCM) of voice frequencies. Recommendation G.711, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, Nov. 1998.

[6] IRT Lab, Columbia University. sipc home page. http://www.cs.columbia.edu/IRT/software/sipc.

[7] Moon, S. B., Kurose, J., and Towsley, D. Packet audio playout delay adjustment algorithms: performance bounds and algorithms. Research report, Department of Computer Science, University of Massachusetts at Amherst, Amherst, Massachusetts, Aug. 1995.

[8] Perkins, C., Hodson, O., and Hardman, V. A survey of packet loss recovery techniques for streaming audio. *IEEE Network 12*, 5 (Sept. 1998), 40–48.

[9] Ramjee, R., Kurose, J., Towsley, D., and Schulzrinne, H. Adaptive playout mechanisms for packetized audio applications in wide-area networks. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)* (Toronto, Canada, June 1994), IEEE Computer Society Press, Los Alamitos, California, pp. 680–688.

[10] Rosenberg, J., and Schulzrinne, H. An RTP payload format for generic forward error correction. RFC 2733, Internet Engineering Task Force, Dec. 1999.

[11] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V. RTP: a transport protocol for real-time applications. RFC 1889, Internet Engineering Task Force, Jan. 1996.

[12] Semiconductor, D. DS1553: 64k NV Y2KC Timekeeping RAM Manual. Tech. rep., Dallas Semiconductor, 1999. At http://pdfserv.maxim-ic.com/arpdf/DS1553-DS1553P.pdf.