QoS Evaluation of VoIP End-points

Wenyu Jiang
Department of Computer Science
Columbia University
Email: wenyu@cs.columbia.edu

Kazuumi Koguchi
Information Technology R&D Center
Mitsubishi Electric Corporation
Email: koguchi@isl.melco.co.jp

Henning Schulzrinne
Department of Computer Science
Columbia University
Email: hgs@cs.columbia.edu

***Abstract*— We evaluate the QoS of a number of VoIP end-points, in terms of mouth-to-ear (M2E) delay, clock skew, silence suppression behavior and robustness to packet loss. Our results show that the M2E delay depends mainly on the receiving end-point. Hardware IP phones, when acting as receivers, usually achieve a low average M2E delay (45-90 ms) under low jitter conditions. Software clients achieve an average M2E delay from 65 ms to over 400 ms, depending on the actual implementation. All tested end-points can compensate for clock skew, although some suffer from 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. All the hardware IP phones we tested support some form of packet loss concealment better than silence substitution. The concealment generally works well for two to three consecutive losses at 20 ms packet intervals, but voice will quickly deteriorate beyond that.**

I. 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. Although there is an extensive literature on network quality of service (QoS) [13], little has been done on studying the QoS of VoIP end-points. Here, we evaluate several well-known brands of IP phones and a few popular software VoIP clients. Our evaluation has revealed quite a few interesting results. For example, Microsoft Messenger achieves the lowest mouth-to-ear (M2E) delay [1] (65-120 ms on average) among the software VoIP clients, while its predecessor NetMeeting is among the worst ($>$ 400 ms). Even more surprisingly, while Messenger XP has been touted as the best Messenger implementation, our experiments show that Messenger on Windows 2000 consistently achieves a lower M2E delay, with exactly the same hardware. Only a few of our end-points support silence suppression, but luckily all end-points are able to adjust playout delay adaptively even when the sender transmits continuously. However, the silence detectors they use may falsely treat music as silence, creating unwanted gaps in the output audio.

The remainder of this paper is organized as follows. Section II lists related work. Section III describes the setup of our experiments. Section IV presents the measurement results, and Section V concludes the paper and lists future work.

II. Related Work

Most QoS technologies aim to reduce network delay, whether through bandwidth reservation in IntServ, prioritized forwarding in DiffServ and MPLS, or congestion avoidance in constraint based routing, all of which are well summarized in [13]. However, few papers have addressed end-to-end (M2E) delay in real applications. We came across one related project by Calyam [2] and Schopis [10] that studies the M2E delay on H.323 videoconferencing systems. They use a metronome (pulses) as an input source and use an oscilloscope to view the phase offset between input and output pulses and derive the M2E delay. Their delay measurement technique is similar to ours, but we use real speech instead of a metronome, and we derive the delay automatically through software-based estimation. The delay of video conferencing systems is expected to be higher than IP phones (voice only) due to potentially long look-ahead delay such as from B-frame (bi-directional) prediction in video coding. Our study evaluates more VoIP products and addresses other metrics including clock skew and silence detector behavior.

Sage Instrument has a few products such as 935AT 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.

III. Experiment Setup

A. Measurement Approach

We measure mouth-to-ear delay by recording both the original and output audio in a two-channel (stereo) mode, as illustrated in Figure 1. The couplers in Figure 1 are special audio feeding/receiving devices that connect between the handset and base of a telephone such as an IP phone.

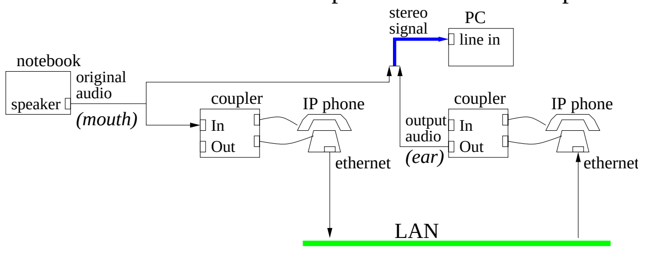


Fig. 1. Measuring mouth-to-ear (M2E) delay

To automate delay estimation, we use the adelay software (http://www.cs.columbia.edu/IRT/software/adelay/) developed in our lab. It reads a stereo audio file and calculates a most likely delay offset based on auto-correlation in the time domain. We have tested its correctness by inserting a known delay offset with a special audio effects unit. The precision of adelay is always within 1 ms in these tests.

We used the digital recording of an audio book tape cassette as the audio source in our experiments.

[1]Whenever we refer to an end-point's M2E delay, we imply that it is acting as a receiver. Unless otherwise noted, we assume a LAN test environment

B. End-point Devices and Configurations

We evaluate IP phones from three major vendors: Cisco, 3Com and Pingtel, plus a 1-line PSTN/IP gateway made by Mediatrix. The software clients include Microsoft Messenger and NetMeeting (both running on a Athlon/K7 900 MHz PC with Windows 2000/XP dual-boot, and a Pentium-3 800 MHz notebook with Windows XP), sipc [5] using rat from UCL (http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/) as the media client, and Net2Phone. Table I lists all tested end-points and whether silence suppression is used.

End-point & client version	platform/model or firmware version	silence suppression
Cisco phone	7960, AppLoad P0S30201	Y
3Com phone	NBX 1.0.1.26.1	N
Pingtel phone	Xpressa 1.2.7.4	N
Mediatrix gateway	APA III 1FXS 1.0.6.12	N
rat 4.2.20	Solaris, Ultra-10 360 MHz	N
Messenger 4.6	Win 2000/XP, K7-900 & P3-800	Y
NetMeeting 3.0	Win 2000/XP, K7-900 & P3-800	Y
Net2Phone 1.1 & 1.5	Win NT P2-450 & Win 98 P3-1G & Win 2000 K7-900	Y
GSM phone	GSM 1900 US	n/a

TABLE I

LIST OF TESTED END-POINTS, FIRST FIVE ARE SIP [9] BASED

Table II shows some of the configuration parameters. Most end-points implement the G.711 μ-law [4] codec, but not many other codecs are supported. Due to interoperability limitations, we cannot test all pairwise combinations of end-points. For example, Net2Phone only talks to other Net2Phone clients.

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 II

COMMON PARAMETERS IN THE EXPERIMENTS

IV. MEASUREMENT RESULTS

A. Mouth-to-ear (M2E) Delay without Jitter

Figure 2(a) shows how M2E delay evolves over time. The notation “experiment 1-2” means part 2 of call 1. Between part 1 and part 2 is a short pause when we save part 1’s audio to disk. In experiment 1-1, the M2E delay from 3Com to Cisco peaks at about 59 ms at the time of 254 seconds, then drops to 49 ms immediately. The highly linear trend shows the effect of clock skew. A similar linear trend and drop in M2E delay (also by 10 ms) are observed in experiment 1-2.

In Figure 2(a) we also draw the long silence gaps ($>$ 1 sec), with height representing relative gap length. There is a clear correspondence in Figure 2(a) between gaps and time-points of delay adjustment. This concurs with the convention that playout delay should only be adjusted at the beginning of a new talk-spurt [7], an event represented by an RTP [11] packet with the marker (M) bit set to 1. However, the 3Com phone does not use silence suppression. Hence, its RTP packets never have the M-bit set. But the Cisco phone is still able to adjust the playout delay irrespective of the M-bit, and the adjustment still occurs during a silence gap.

Figure 2(b) shows how the presence of M-bits (Cisco to 3Com) serves as good hints for delay adjustment in the receiver. In contrast, delay adjustment occurs much less often and the M2E delay exhibits a cleaner linear trend when no M-bits are sent, such as in the Pingtel to 3Com case.

We notice that certain senders can introduce delay spikes. Figure 2(c) shows many small spikes, plus two high spikes ($>$ 200 ms) that causes distortions in speech. Given our test LAN is lightly loaded with virtually no jitter, and the spikes occur only when Mediatrix is the sender, the Mediatrix gateway has to be the source of the problem. We conjecture that it has a poor real-time scheduler, which would also explain our observation of similar spikes even when it is the receiver.

sender	receiver				
	3Com	Cisco	Mediatrix	Pingtel	rat
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)
w. G.729		98.7 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)
rat	52.4 ms (3.1 ms)	59.8 ms (0.9 ms)		64.2 ms (16.4 ms)	

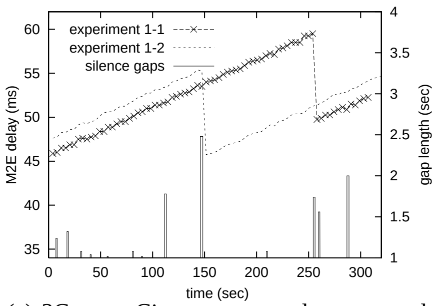
TABLE III

AVERAGE M2E DELAYS FOR IP PHONES AND rat

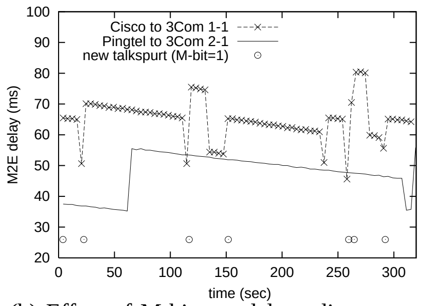
Instead of showing all test pairs’ delay plots, Table III summarizes them with the average M2E delay. Its first column represents the sender, and its first row denotes the receiver. The numbers in parentheses are the difference (range) between the highest and lowest average M2E delay among all calls for the same pair of end-points. A low range indicates high repeatability, which is confirmed by Table III 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 using G.729 [3] is 98.7 ms, nearly 15 ms higher than for G.711 (75.8 ms), clearly due to additional compression and look-ahead 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 rat is the receiver. Its M2E delay is consistently above 200 ms irrespective of the sender.

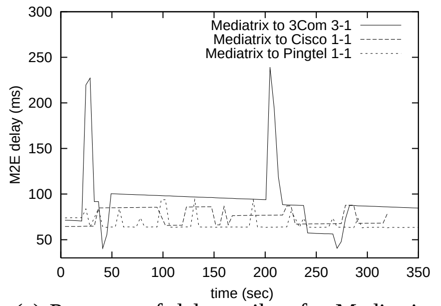
The dominant role of the receiver is more evident in Table IV. For example, Messenger XP (notebook) to XP (pc) achieves an average delay of 120 ms. However, when the receiver switches to Messenger 2000 (same PC), the delay consistently drops to 68.5 ms, indicated by the small range in Table IV. This 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,



(a) 3Com to Cisco; gaps *not* drawn to scale



(b) Effect of M-bits on delay adjustment



(c) Presence of delay spikes for Mediatrix

Fig. 2. Mouth-to-ear (M2E) delay plots

while we find that Messenger 2000 actually has lower delay, with exactly the same hardware.

The sender does in some cases play a minor role in M2E delay. When Messenger XP (notebook) 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. This could come from a reduction of the processing time required by Messenger before sending out every packet.

end-point A	end-point B	A → B	B ← A
MgrXP (pc) (range)	MgrXP (notebook) (Mgr: Messenger)	109 ms (0.8 ms)	120 ms (44.6 ms)
Mgr2K (pc)		96.8 ms (5 ms)	68.5 ms (10.1 ms)
NM2K (pc)	NMXP (notebook) (NM: NetMeeting)	401 ms (46.9 ms)	421 ms (6.8 ms)
Net2Phone v1.1, NT P-2	PSTN (local number)	292 ms (77.3 ms)	372 ms (11.1 ms)
Net2Phone v1.5, NT P-2		201 ms (2.7 ms)	373 ms (9.2 ms)
Net2Phone v1.5, W2K K7		196 ms (9.8 ms)	401 ms (2.1 ms)
Net2Phone v1.5, W2K K7	Net2Phone v1.5, W98 P-3	525 ms (2.7 ms)	350 ms (8 ms)
Mobile (GSM)	PSTN (local number)	115 ms (4.8 ms)	109 ms (5.1 ms)

TABLE IV
AVERAGE M2E DELAYS FOR PC CLIENTS AND MOBILE PHONE

The predecessor of Messenger is NetMeeting, and its M2E delay is a very high 400 ms, using the same PC and notebook. This is quantitative evidence that NetMeeting is not well designed for real-time interactive conversations, and the delay bottleneck is the application rather than the OS in use.

The last VoIP end-point we test is the Net2Phone CommCenter, a popular IP telephony application. Its M2E delay is unfortunately also quite high, near or above 300 ms with v1.1. The round-trip time between the NT PC and its PSTN gateway is usually 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.

Later we upgraded the Net2Phone software on the NT machine to v1.5. The only main difference is the average delay from Net2Phone to PSTN dropped from 292 ms to 201 ms. It indicates the sender (Net2Phone) also introduces some noticeable delay. The results on the Windows 2000 PC (same one used for Messenger) are similar to that of NT, even though the hardware is quite different. The delay between a PC-to-PC call is, however, much higher in the Win2000 to Win98 case (525 ms) than Win2000 to PSTN (196 ms), suggesting the Win98 notebook introduces a lot of delay. This again shows the receiver plays a more important role.

With a black-box measurement approach, we cannot know the exact bottleneck for high delays. However, we can name the likely candidates. This includes the playout delay, sound API, sound card driver, OS, and the sound card itself.

Finally, it motivates to compare VoIP with mobile phones, here Messenger is comparable to GSM in delay (≈110 ms).

B. Clock Skew

According to Mills' notation, we define clock skew as the frequency difference [6]. In our experiments, all the end-points adjust the playout delay to deal with clock skew, such as in Figure 2(a). The 3Com and Pingtel phone appear to occasionally suffer from playout buffer underflow even when there is no packet loss or delay jitter. Figure 3 shows the input (top) and output (bottom) waveforms, and highlights when a waveform drop occurs (3:57.759 ≈ 238 sec). It correlates well with Figure 2(b), Cisco to 3Com 1-1 case, exactly when the delay is being adjusted downward then upward. It is therefore most likely a side effect (or bug) of the playout algorithm.

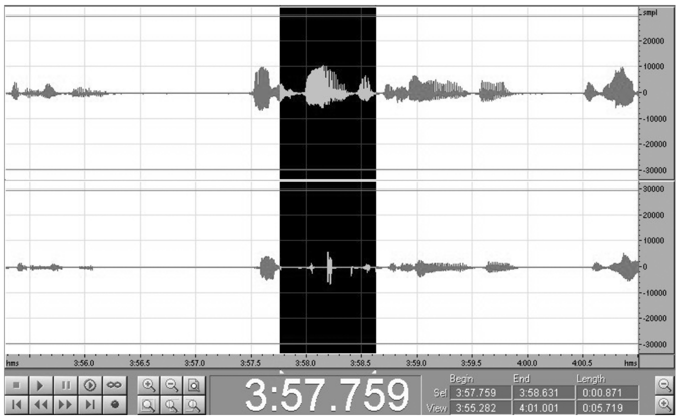


Fig. 3. Playout buffer underflow, Cisco to 3Com 1-1; dropped waveform

Next we analyze the rate of clock skew. If the skew 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 skew is symmetric, that is, the skew rate from B to A should be -r. This can be seen in Table V.

The skew rate depends on the crystal oscillator used to drive the voice circuitry, but its magnitude often falls within 25 ppm

[12] and rarely exceeds 100 ppm. Between the IP phones, the magnitude of the combined skew is within 60 ppm. Between two Cisco phones, the skew is almost zero, probably because they use the same type of oscillators. However, for rat it is always higher than 300 ppm, far beyond 100 ppm. We have measured other workstations and PCs, and found many have skew rates of 200-300 ppm.

sender	receiver				
	3Com	Cisco	Mediatrix	Pingtel	rat
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
rat	343	403		376	
Cisco G.729		-0.4			

TABLE V

AVERAGE CLOCK SKEW RATES (IN PPM) FOR IP PHONES AND rat

Table VI summarizes the skew rates for PC clients. Surprisingly, Messenger has a positive skew rate in either direction, but interestingly, the skew in A $\rightarrow$ B direction is about twice that of B$\leftarrow$A. NetMeeting exhibits highly oscillative delays, as a result its values are not reliable. The skew rates of Net2Phone on the NT machine are very symmetric, and relatively high, nearly 300 ppm and quite close to that of rat. The skew rates of Net2Phone on the Win2K PC are very close to that of Messenger on the same Win2K PC, and they are both positive. Such a high level of correspondence indicate these results are unusual but real. One logical explanation is that the sound card may have two circuit-driving oscillators. Finally, the mobile phone exhibits no sign of skew, because it is known that the mobile phone network and the PSTN share a common, global stratum-1 clock.

end-point A	end-point B	A $\rightarrow$ B	B $\leftarrow$ A
MgrXP (pc)	MgrXP (notebook)	172	87.7
Mgr2K (pc)		165	85.6
NM2K (pc)	NMXP (notebook)	?	-33?
Net2Phone NT	PSTN	290	-287
Net2Phone W2K		166	82
Mobile (GSM)		0	0

TABLE VI

AVERAGE SKEW RATES (IN PPM) FOR PC CLIENTS AND MOBILE PHONE

C. Behavior under Packet Loss

Packet loss concealment (PLC) [8] ranges from silence substitution (worst quality), packet repetition, to extrapolation and interpolation. We evaluated PLC behavior on the Cisco, 3Com and Pingtel phones, by inserting deterministic bursty losses. In our experiments, 5/100, for example, means 5 consecutive losses every 100 packets. We find that first, all phones implement a PLC better than silence substitution. Second, their PLCs work for up to two (three for Cisco) consecutive losses at a 20 ms interval, then the voice quickly fades to silence. This is acceptable because repairing beyond three consecutive losses becomes difficult and it may cause side effects like buzzing sound. Thirdly, PLC does not affirmatively increase the M2E delay.

By examining the output waveform, we find that the Pingtel phone repeats the waveform, but it smoothes any discontinuity between frames. Both Cisco and 3Com phones use at least extrapolation. Interpolation works if the loss gap is shorter than the playout delay, but we cannot know whether the Cisco and 3Com phone opt to do so just by examining the waveform. Instead we estimate the PLC methods indirectly by comparing their relative audio quality, as shown in Table VII. The quality rating is subjective, but the difference between the three levels (almost inaudible, audible and very audible) are clear-cut. The Cisco phone is the most robust. In particular, for loss pattern 1/20 (effectively 5% loss), it still has good quality, while the other phones perform noticeably worse. In a 1/20 loss pattern, the loss gap (20 ms) is very likely to be less than the playout delay, which would allow interpolation. However, the fact that the 3Com phone has a much worse quality suggests it is not doing interpolation even for the 1/20 case. Therefore, in Table VII we tentatively mark it as extrapolation only. The Pingtel phone performs about the same as 3Com, except that its packet repetition occasionally introduces repetitive or buzzing sound. Finally, when the loss pattern is 1/100, there is no audible distortion for any of these IP phones.

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

TABLE VII

PLC BEHAVIORS OF IP PHONES, G.711 μ-LAW, 20 MS INTERVAL

In terms of latency, we find that PLC does not introduce any consistent and extra delay in any of the three phones, so for brevity we do not show any more delay plots.

D. Silence Suppression Behavior

1) Hangover Time: It is defined as 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 fixed, a value of 200 ms is usually sufficient [1].

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 VIII

SILENCE DETECTOR PROPERTIES

Because the end-points' hangover times were not available to us, we measured their values using waveforms consisting of several on (sine wave) and off (silence) periods. Table VIII summarizes our results. The Cisco phone has a fairly long hangover time. NetMeeting's is the shortest, but still longer than the 200 ms used in [1]. The Cisco phone generates comfort noise packets as well. 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.

2) Robustness for Non-speech Signals (Music): We find that none of the three silence detectors in Table VIII work 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 can still fail when the input level is low.

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 IX

SILENCE DETECTOR PERFORMANCE FOR MUSIC (2.5 MINUTES LONG)

Table IX shows the total number and duration of unnatural gaps created by the Cisco phone and Messenger 2000, at various audio input levels. The input level here is defined as the average volume of the entire music track. Rather than relying on analog measurement, which depends on both electrical current and impedance, we measure the input level in its digital form as observed by the client, by LAN-sniffing the transmitted RTP [11] payload. Messenger clearly uses a higher detection threshold than the Cisco phone. Generally, a silence detector should work for speech level from -36 dB to -26 dB or higher. 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 well for speech, it may not for music.

E. M2E Delay under Network Jitter

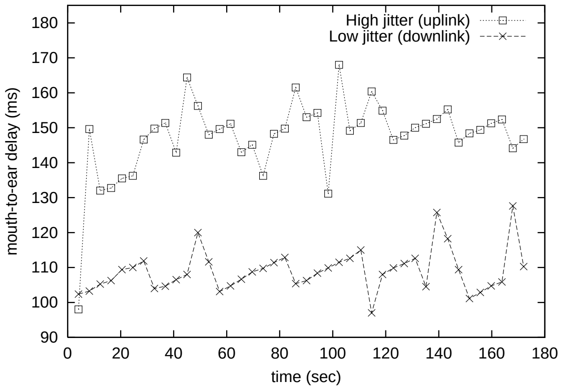


Fig. 4. Impact of jitter on M2E delay, rat to Cisco, cable modem trace

Finally, we performed an initial test on how jitter affects the M2E delay. We used a typical packet trace collected between a university machine and a PC with cable modem. It is well known that cable modem uplink exhibits high jitter. The 99% network delay is 43.8 ms for the downlink trace we used, and 93.6 ms for the uplink trace, nearly 50 ms higher. Figure 4 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. In addition, we do not notice any distortion in the output audio. Therefore, the Cisco phone does a good job of playout buffering and suffers no audible late loss.

V. CONCLUSIONS AND FUTURE WORK

We performed a large number of measurements on various VoIP end-points. All the hardware IP phones achieve a low M2E delay and have acceptable packet loss concealment performance, with the Cisco phone being the most robust. Among software-based clients, Messenger 2000 has the lowest average M2E delay (68.5 ms), even better than Messenger XP (97-120 ms). rat achieves 200-230 ms in comparison. NetMeeting performs almost the worst, with over 400 ms on average. The delay of Net2Phone range from 200 ms to 525 ms, depending on the software version, the direction, and OS.

The magnitude of clock skew is within 60 ppm between IP phones, but can be quite high ($>$ 280 ppm) for rat (running on a workstation) and Net2Phone (running on a PC).

Silence detectors used by the Cisco phone, Messenger and NetMeeting 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.

Finally, a 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. We plan to conduct similar experiments on other end-points and with more jitter conditions, and we will examine other important metrics such as echo cancellation.

REFERENCES

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

[2] Anjaneya Prasad Calyam. Performance measurement and analysis of H.323 videoconference traffic. Master's thesis, Department of Electrical Engineering, Ohio State University, June 2002.

[3] International Telecommunication Union. Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction. Recommendation G.729, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1996.

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

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

[6] D. L. Mills. Network time protocol (version 3) specification, implementation. RFC 1305, Internet Engineering Task Force, March 1992.

[7] Sue B. Moon, James F. Kurose, and Donald F. Towsley. Packet audio playout delay adjustment algorithms: performance bounds and algorithms. Research report, Department of Computer Science, University of Massachusetts at Amherst, Amherst, Massachusetts, August 1995.

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

[9] J. Rosenberg, Henning Schulzrinne, G. Camarillo, A. R. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: session initiation protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[10] Paul Schopis. Summary test H.323 bounds test report. Technical report, Internet2 Technology Evaluation Center, Ohio, August 2001.

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

[12] Dallas Semiconductor. DS1553: 64k NV Y2KC Timekeeping RAM Manual. Technical report, Dallas Semiconductor, 1999.

[13] Xiaojun Xiao and L. M. Ni. Internet qos: A big picture. *IEEE Network*, 13(2):8–18, March/April 1999.