Simulating Playout Schedulers for VoIP - Software Manual

Christian Hoene, Sven Wiethölter
hoene@ieee.org
TKN, Technical University of Berlin, Germany

27th September 2004

Abstract

This document describes an open-source software package, which evaluates both experimental and simulated VoIP systems. It includes various playout schedulers for Voice over IP (VoIP) described by Schulrzinne, Ramjee, Moon, and others. The two most common speech encoding schemes (G.711 and G.729) are also supported. Two quality models (ITU's PESQ and E-Model) predict the perceptual end-to-end transmission quality. Finally, an add-on to the discrete event simulator ns-2 is described with which packet traces for the quality assessment can be produced.

1 Introduction

Perceptual quality assessment has to take into account the complete end-to-end transmission path because it reflects human-to-human conversation, starting at the mouth of the talker until the acoustic signal reaches the ear of the listener [9]. Thus, if a transmission of VoIP is to be studied, anyhow the entire transmission system has to be considered. The end-to-end quality depends largely on the playout buffer scheme [4]. Playout buffers temporarily store packets at the receiver to play them out in a timely manner.

Annoyingly, until now it has not been agreed upon a common, standardized playout buffer scheme. Instead, one is free to chose any scheme for his system implementation. Therefore we have to assume the presence of multiple difference adaptive and fixed playout buffer schemes, each with a different parameter set. In order to consider multiple playout buffer schemes, it is important to cover the entire known space of implementations.

This software package contains the most well known playout schedulers. We reimplemented the algorithms describe by Schulzrinne, Ramjee, Moon and Lin. We also include perceptual assessment tools and algorithms called PESQ and the ITU E-Model. This software encodes a speech sample, applys a given trace of VoIP packets, simulates multiple playout schedulers, and finally assesses the quality of telephone services (packet loss, transmission delay and playout rescheduling). Thus, it can determine the final packet loss rate, speech quality, mean transmission delay and conversational call quality.

1.1 Features

The software described in this manual covers the entire processing chain of VoIP. It supports the

- encoding of audio samples with G.711 μLAW and G.729.
- parsing of packet trace files generated by Snuffle [7] or NS/2 to gather data about a packet's delay or loss.
- generating packet trace files artifically to study the impact of delay spikes.

- playout scheduling as describe by Moon, Ramjee and others.
- decoding of audio file, including the rescheduling of the playout time.
- quality assessment using ITU's E-Model and PESQ.

1.2 Description

Playout buffers adapt the playout time during the transmission. This rescheduling might harm the speech quality because of temporal discontinuities. However, the E-Model does not take into account the dynamics of a transmission but relies on static transmission parameters. PESQ instead considers playout adaption but does not include the absolute delay into its rating. Therefore we combine both models. PESQ calculates the speech quality and feeds the MOS value into the E-Model. To combine both models mathematically, we applied the formula given in [8] (figure 1).

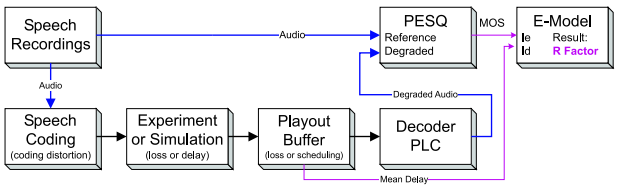

Figure 1: Speech and Delay Assessment

1.3 Background: Voice over IP

Internet Telephony allows to offer telephony services across networks using the Internet protocols and is an alternative to the classic telephone system (PSTN). The principle components of the of VoIP system, which covers the end-to-end transmission of voice, are displayed in figure 2.

First, digitized human speech is encoded. Encoding algorithms compress audio signals. Most speech encoding schemes compress segments of speech and generate frames. One or multiple speech frames are concatenated in one packet. RTP, UDP, and IP

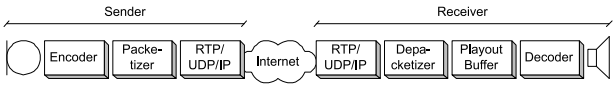

Figure 2: VoIP Transmission Path

packet headers are added to the speech segments, before the packets are send the to receiver. The network transmits packets from the sender to the receiver. In the Internet packets get lost because of congestion or (wireless) transmission errors. The transmission delay of packets, the time needed to transmit a packet from the sender to the receiver, is variable and depends on the current network condition and the routing path [10, 4, 6, 5].

At the receiver, protocols process the packets and deliver them to the de-jittering buffer which temporally stores packets so that they can be played out in a timely manner. If packets are too late to be played out on time, they are usually regarded as lost. A couple of publications study [16, 17, 15, 18, 11], how to choose the ideal time of playing out the received frame. The size of the dejitter buffer should be adjusted so that most packet are not received too late and packet losses are minimized. Futhermore dejitter buffer for VoIP should be adapted immediately to avoid an increase in the transmission delay. The schedule of playout can be adjusted most easily during silence because then it is not notable. Adjustments during voice activity require more sophisticated concealment algorithms [12, 13]. In summary, the losses from the application point of view are in fact a superposition of real losses and excessive delays, whereas excessive delay is used in terms of play-out buffer dimensioning (non-standardized algorithms).

As the next step, the speech frames are decoded. If a frame is lost, the decoder will conceal the lost frame and extrapolates the last successfully received frame [14]. Finally, the digital signal is transformed to an acoustic signal.

1.4 Background: Quality Assessment

On the other hand, VoIP applications cannot measure subjective impressions but packet loss rates,

round trip times, and packet delay distributions. However, network- and transport-layer metrics do not reflect the perceived quality precisely. In the last year, considerable efforts have been made to predict human rating behaviour using precisely measureable parameters. We will describe shortly the most common quality models for telephony:

The Perceptual Assessment of Speech Quality (*PESQ*) algorithm predicts the speech quality of narrow band speech transmission. The PESQ algorithm is standardized in [3]. It compares the original and the degraded version of the sample to assess the speech quality with a mean opinion score value (MOS), which scales from 1 (bad) to 5 (excellent).

The quality of a telephone call is not judged by the speech quality entirely. Further factors have to be considered. The ITU *E-Model* [2] takes into account various other impairments like delay and echoes to calculate the R-factor. A higher R factor corresponds to a better telephone quality, zero being the worst value, 70 toll quality, and 100 excellent quality. One novel feature of the E-Model is the assumption that sources of impairment, which are not correlated, can be added on a psychological scale. This allows to trade off between different sources of impairment (e.g. loss versus delay).

2 Usage

This software has been developed and tested under Linux. It is command-line based. To call it, type

```
> cd playout/examples
> ./playout <parameters> <source file>
```

The parameter <source file> describes an audio file, which will be evaluated. The playout program supports the parameters in the following table.

The required input format of the file has to be 16bit PCM encoding, 8000 Hz sampling rate, mono, littleendian byte order. To convert a file to the required input format call the following program.

```
> sox <input> -r 8000 -c 1 -sw <output>
```

If the program is called, it generates a couple of results, which are

- the console output contains quite a lot information about the the compressing process, playout scheduling and quality assessment. For verification purpose it might be saved.
- the file "playout_result.txt" is being extended by the results of the playout scheduler simulation and assessment, which are the mean delay, MOS (between 1 and 5), R factor (between 0 and 100), the audio file produced (if any), the loss rate (between 0-1), the number of lost frames and the number of lost frames due to network losses.
- a couple of "resultXXX.sw" files contain the transmitted audio content, including the distortion due to coding, frame loss and playout rescheduling. They can be used for auditory verifications.

2.1 Usage Examples

Let us a assume that we like to assess the transmissions of a VoIP stream. The VoIP packet trace is given in an RXStat file. Then we call

```
> ./playout -c G711 -t
  stations10_rxstats_w8_wl8.out
  -d 0.150 -f 0.01 -m -r 3-a_f01s13.sw
```

In the above example we assume a system delay, which is add to the packet transmission delay and playout delay, of 150ms. Further we choose multiple fixed length playout buffers with an intermediate step of 10ms. We want to simulate Moon's and Ramjee's algorithms. The result is appended to the "playout_result.txt" file in which the audio results are stored too.

If we consider the impact of so called delay spikes, the program can also simulate the trace of VoIP packets. E.g. for having five uniform delay spikes in the audio file, each spike with a length of 200ms and a height of 100ms, call

```
> ./playout -c G711 -T
  $RANDOM,5,0.1,0.2 -d 0.150 -S
```

Parameter	Description	Default
-c [G711|G729]	Select the coding scheme which is used for simulation.	G711
-t <tracefile>	Read a trace file containing packet delays.The format of the trace file has to be RXSTAT (see appendix)	n/a
-T <seed>,<NoOfSpike>, <Height>,<Weight>	Generate a trace file containing delay spikes	n/a
-d <system delay>	Adds the end-to-end delay of the transmission minus the delay introduce by the playout buffer. Value has to be between 0s and 1s	0.15s
-o <trace offset>	Causes the program to drop the beginning of the packet trace file.	0s
-f <step offset>	Simulate multiple fix-deadline playout buffers. Starting at multiple of <step offset>. Increasing the deadline in <step offset>s steps.	n/a
-F <step offset>	Dito. But starts which the transmission delay of the fastest packet.	n/a
-m	Simulate Moon's algorithms.	see source code
-r	Simulate Ramjee's playout schedulers (7 different parameter sets).	see source code
-S	A playout scheduler that follows each delay spike. If a period of silence is reached the playout time falls back to the normal level.	n/a

Table 1: Command line parameters

The position of the delay spikes are forced to be in an area with voice activity (no spikes during silence). The actual position depends on the first parameter, which is a random seed value. One should note that the width of a delay spike SHOULD be larger than the weight, otherwise the behavior is undefined. The parameter -S schedules the playout with three different algorithms. First, it drops all packet which are delayed by the spike. Second, it delays the playout by the height of the spike. Last, it delays the playout by the height of the spike but reschedulse the playout next time a talkspurt begins.

3 Compiling and Installation

This program runs under Linux. It has been tested on a Debian and Suse 9.1 distribution. It has been developed and compiled with the development studio "KDevelop 2.1". To compile open the KDevelop project file "playout2.kdevprj" or call

```
> cd playout
> ./configure
> ./make
```

4 VoIP simulations with ns-2.26

To gain the appropriate VoIP packet traces for the evaluation with the playout tool, we wanted to conduct DES simulations by handing predefined input data to the simulation instance.

We used the popular simulation tool "network simulator ns-2" which supports wired and wireless networking protocols. The simulator is an open-source project and its code is available on the Internet [1].

The development of a simulation model in ns-2.26 which consists of a VoIP application and an enhanced RTP agent was necessary because ns has not supported further processing of bit stream or packet traces yet. Additionally, one is bound to the standard ns traces which do not include sufficient information for the playout tool.

We provide our VoIP related changes to ns-2, which are included in the archive ns-VoIP.tgz, as open-source software.

4.1 VoIP Application

Coded voice data which was converted into a bit stream is handed to the VoIP application. During simulation, the application works off the bit stream file at the sender side and builds audio frames according to the predefined settings (generation interval and packet size).

On the receiver side, the application keeps track of arrived packets and dumps out packet's information about generation time, arrival time, sequence number, version and marking flags into the output trace file. This file can be given to the playout tool which finally assesses the quality of the VoIP flow.

The application can represent either a sender or a receiver process as well as a combination of both. The source code of the VoIP application can be found in the files voip_traffic.cc / .h.

4.2 Advanced RTP agent

The standard ns RTP agent is only able to generate packets its own. We enhanced the basic RTP agent because we needed an instance which handed the VoIP packets of the application downwards. Basically, we introduced interfaces to the above described application process. The implemtation is included in rtpdata.cc / .h.

4.3 Installation

For the installation of the described application and the RTP agent, lots of changes in ns are necessary. Please follow the description in the readme file with is part of the archive ns-VoIP.tgz.

5 Concluding Remarks

This software package is entirely open source exept the PESQ and G.729 program. These are covered by rights of their owners. However, you can download the for free from the web site of ITU. If you have any suggestion on how to improve this software packet, please contact the author.

References

[1] The network simulator ns-2. URL: http://www.isi.edu/nsnam.

[2] Itu-t. recommendation g.107. the e-model, a computational model for use in transmission planning, 5 2000.

[3] 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, 2 2001.

[4] Fouad A. Tobagi Athina P. Markopoulou and Mansour J. Karam. Assessment of voip quality over internet backbones. Infocom 2002, 2002.

[5] J. Bolot. Characterizing end-to-end packet delay and loss in the internet, 1993.

[6] Jean-Chrysostome Bolot, Hugues Crepin, and Andreas Vega Garcia. Analysis of audio packet loss in the internet. In *Network and Operating System Support for Digital Audio and Video*, pages 154–165, 1995.

[7] C. Hoene. Easysnuffle - a tool to measure the performance of multimedia flows over ieee 802.11b. URL: http://www.tkn.tu-berlin.de/research/easysnuffle/, March 2001.

[8] C. Hoene, B. Rathke, and A. Wolisz. On the importance of a voip packet. In *Proc. of ISCA Tutorial and Research Workshop on th Auditory Quality of Systems*, Mont-Cenis,Germany, April 2003.

[9] C. Hoene, S. Wiethölter, and A. Wolisz. Predicting the perceptual service quality using a trace of voip packets. In *Proceedings of Fifth International Workshop on Quality of future Internet Services (QofIS'04)*, Barcelona, Spain, September 2004. To appear.

[10] Ingemar Kaj and Ian Marsh. Modelling the arrival process for packet audio. In *Quality of Service in Multiservice IP Networks*, pages 35–49, Milan, Italy, February 2003.

[11] Nikolaos Laoutaris and Ioannis Stavrakakis. Intrastream synchronization for continuous media streams: A survey of playout schedulers. *IEEE Network Magazine*, 16(3), May 2002.

[12] Y. J. Liang, N. Färber, and B. Girod. Adaptive playout scheduling and loss concealment for voice communication over ip networks. *IEEE Transactions on Multimedia*, 5(4):532–543, 12 2003.

[13] F. Liu, J. Kim, and C.-C. J. Kuo. Adaptive delay concealment for internet voice applications with packet-based time-scale modification. In *in Proc. IEEE ICASSP*, May 2001.

[14] C. Perkins, O. Hodson, and V. Hardman. A survey of packet loss recovery techniques for streaming audio. *IEEE Network*, 12:40–48, Sep/Oct 1998.

[15] Jesus Pinto and Kenneth J. Christensen. An algorithm for playout of packet voice based on adaptive adjustment of talkspurt silence periods. In *LCN*, pages 224–231, 1999.

[16] Ramachandran Ramjee, James F. Kurose, Donald F. Towsley, and Henning Schulzrinne. Adaptive playout mechanisms for packetized audio applications in wide-area networks. In *INFOCOM (2)*, pages 680–688, 1994.

[17] D. Towsley S. B. Moon, J. Kurose. Packet audio playout delay adjustments: perforamnce bounds and algorithms. *ACM/Springer Multimedia Systems*, 27(3):17–28, 1 1998.

[18] C.J. Sreenan, J.-C. Chen, P. Agrawal, and B. Narendran. Delay reduction techniques for playout buffering. *IEEE Transactions on Multimedia*, 2(2):88–100, June 2000.

Date	**Comments**
15th December 2003	First release
27th September 2004	Correction of the manual, added ns/2 module, added VoIP application source

Table 2: Version History