AN INTEGRATED NTP-RTCP SOLUTION TO AUDIO SKEW DETECTION AND COMPENSATION FOR VOIP APPLICATIONS

Hugh Melvin, Liam Murphy

Department of Computer Science
University College Dublin, Ireland

ABSTRACT

The circuit switched POTS (Plain Old Telephone System) preserves the timing relationship between media samples from sender to receiver through use of a common clock. For PC-based Internet multimedia, the existence of separate audio and system clocks on either end-host can introduce significant complications. Much work has taken place in recent years that addresses the issue of system clock skew and its effect on precise delay measurement. In a Voice over IP (VoIP) environment, where adaptive buffering techniques are employed, system and audio clock skew can distort both delay measurement and playout control as well as lead to poor buffer performance. This paper presents a high level mechanism to measure and compensate for the skew relationships between system and audio clocks at each end of a multimedia session. The mechanism utilises both the Network Time Protocol (NTP) and the RTCP (Real-time Transport Control Protocol) Control Protocol. Preliminary and positive results are presented from a testbed system and plans for further work are outlined.

1. INTRODUCTION

The POTS preserves the timing relationships in speech between sender and receiver through use of a common precise clock. The 125 microsecond sample period, fundamental to the POTS is thus implemented at sender and receiver digital exchanges without any skew issues. The trend towards Internet based multimedia in recent years has introduced a range of complexities resulting from the lack of a common clock. A standard PC will contain a number of low-grade oscillator crystals, among them being the system clock to maintain system time, and an audio clock, to set the sample periods for recording/playback etc. Such oscillator crystals can have inherent frequency errors greater than a few hundred parts-per-million resulting in accumulated errors of tens of seconds per day. Fig. 1 illustrates a typical PC system clock frequency error of 18ppm reported in earlier work [1].

Significant work has taken place to detect and correct system clock skew, principally to ensure the accuracy of delay measurements over long periods. Moon et al employ a linear programming approach to the problem [2] whereas Zhang et al outline a convex-hull approach [3].

For VoIP, audio card clock skew raises a number of issues. It can result in poor performance of the receiver playout buffer, introduced to absorb the effects of network jitter. A faster sender audio clock will lead to packet accumulation in the buffer with resulting higher buffer residency delays and buffer overflow (packet loss), whereas a slower sender audio clock will result in buffer underfill. Hodson et al [4] outline a solution to this problem that inserts/deletes appropriate samples to compensate for such skew whereas Akester et al attempt to match the receiver audio clock rate to that of the sender [5]. Both of these approaches utilise a low level mechanism that measures audio skew by monitoring the data flow through the end-device buffers.

In previous work [1] [7], the authors describe a hybrid receiver-based playout algorithm that optimises playout quality by alternating between fixed and adaptive playout delays. In common with most other adaptive playout strategies, this involves utilising the system clock to timestamp incoming packets for delay measurement and playout control. In contrast to the other approaches, the hybrid algorithm utilises synchronised time, achieved through use of the NTP and thus measures absolute end-to-end delays rather than relative delay and network jitter. Figure 2 illustrates the relationship between audio and system clocks for a unicast multimedia session. If the sender audio clock rate (which determines the rate that packets are sent) is different from the receiver system clock (which timestamps packet arrivals), this will manifest itself in an apparent gradual increase or decrease in one-way delay, which will corrupt the performance of all such adaptive playout algorithms. Additionally, if the receiver audio clock rate differs from the receiver system clock rate, further playout complications will occur. As such, there are four separate clocks contributing to the session, each with its unique frequency characteristics. Through novel use of timestamps within the RTCP protocol, coupled to the existing NTP protocol, all four clock characteristics can easily be determined.

The remainder of the paper is organised as follows. Section

Fig. 1. Measured PC System Clock Drift

Fig. 2. Audio and System Clocks
II outlines the mechanism for measuring skew through integration of NTP with the RTCP Sender Report (SR) packets. Section III describes the testbed developed to evaluate the performance of the mechanism and provides preliminary results as well as outlining a number of implementation issues. Section IV outlines how this mechanism can be integrated into a VoIP application. Section V concludes the paper.

2. SKEW DETECTION MECHANISM

The Realtime Transport Protocol (RTP) for delivering Internet multimedia includes a timestamp that enables a receiver to accurately reconstruct media packets for playout. The timestamps are media specific and relate to the sample number generated by the codec. The RTP Control Protocol (RTCP) is a companion protocol to RTP. Amongst other things, it provides feedback to the sender relating to the quality of the session as seen by the receiver. The RTCP SR packets also include two timestamps that are intended to enable a receiver to provide lip-synch and the sender to calculate the round trip time. The timestamps are the system clock timestamp (in NTP format) indicating when the SR packet was generated, along with the corresponding RTP timestamp which is set by the audio card rate. During the lifetime of a media session, each sender periodically generates RTCP SR packets and sends them to each receiving host [6]. If both system and audio clocks are running at exactly the same rate on a given host, the interval between successive RTCP SR packets as indicated by the increment in RTP and NTP timestamps over the period will be equal (e.g., if interval is 10 sec according to the NTP timestamps, the RTP timestamp increment should be 80000, presuming that the sample interval is 125 microseconds). Any difference will indicate the skew between audio and system clock rates. By accumulating timestamp information from successive RTCP SR packets, each receiver can precisely and quickly determine the skew value between the sender’s system and audio clocks.

If additionally, system clocks are synchronised via NTP, this value also represents the skew between sender audio and the receiver system clock. Previous work by the authors indicates that NTP will provide millisecond-level synchronisation on Local Area Networks and well provisioned Wide Area Networks [7]. This combination of RTCP and NTP enables each receiver to determine precisely what compensating factor needs to be applied to incoming packets to avoid the gradual distortion of one-way delay that otherwise will corrupt the performance of adaptive playout algorithms. Furthermore, by examining its own RTCP SR packets being generated for transmission, the receiver can determine the skew between its own audio and system clocks. From an analysis of successive RTCP packets (incoming and generated), each receiver can therefore generate a precise picture of all four clock rates and implement appropriate compensatory action.

3. TESTBED AND RESULTS

Fig. 2 outlines the testbed developed for this work. To evaluate and prove the skew detection mechanism, the authors carried out a series of tests. Implementation details are as follows:

1. The openh323.org code was modified extensively (www.openh323.org). Firstly, incoming packet details were extracted, principally the arrival time (according to receiver system clock) and RTP timestamp (from sender audio clock).

Secondly RTCP SR packets were captured and NTP and RTP timestamps extracted. This enabled the authors to verify that with system clocks synchronised via NTP, the skew determined from successive incoming RTCP packets (sender system versus audio clocks) was reproduced by examining the actual skew of incoming packets as seen by the receiver system clock. Code modifications were required to enforce the atomicity of NTP and RTP timestamp generation.

2. Both end hosts in Fig. 2 were running the Linux OS. Sound card drivers were not modified. Tests were carried out across a lightly loaded LAN to ensure that the packet end-to-end delay experienced minimal jitter. This minimised any noise that may otherwise have distorted the results from (1) above making validation more difficult.

3. A stratum 1 NTP server located on the same network as the two end hosts provided a robust NTP connection for system clocks. Tests were initially carried out with system clocks synchronised via NTP. A second series of tests were then carried out with the host bibio2 freewheeling i.e. running at its natural frequency. This was done to test the consistency of results.

3.1. Results

Results to date have been very positive. Firstly, the effective performance of NTP in synchronising system clocks is critical to the skew measurement mechanism. Fig. 3 shows the performance of one of the hosts over a 36 hour period, during which tests were carried out. The clock offset (relative to UTC) remained within a +/- 1 msec band for 97% of the time. As expected both end-hosts system clocks remained tightly coupled to the NULG NTP server. Full details of NTP performance are presented in [7].

With NTP running on both end-hosts, numerous tests were carried out. A sample set are presented as follows:

Fig. 4 indicates the skew between the audio clock on bibio and the system clock on bibio2 as determined from RTP packets sent from bibio to bibio2. Fig. 5 indicates the skew between the audio and system clocks on bibio as determined from incoming RTCP SR packets received by bibio2.

Fig. 6 indicates the skew between the audio clock on bibio2 and the system clock on bibio as determined from RTP packets...
sent from bibio2 to bibio. Fig. 7 indicates the skew between the audio and system clocks on bibio2 as determined from incoming RTCP SR packets received by bibio2. As evident from the graphs, there is very strong agreement between Fig. 4 and Fig. 5, and between Fig. 6 and Fig. 7. Fig. 8 summarises these results for clarity. As such, a simple analysis of both incoming and outgoing RTCP SR packets thus yields a precise picture of all four clock rates. The mechanism is unaffected by network jitter which can seriously distort the performance of other skew detection mechanisms unless correctly filtered out. Fig. 9 indicates results achieved when only bibio was synchronised via NTP and bibio2 system clock was running at its default (undisciplined) frequency. Although the respective positions of the audio clocks remain constant, the position of bibio2 system clock changes, reflecting its default frequency.

Notwithstanding the above results, a number of software issues relating to OS and sound card driver design require further analysis. Firstly, the RTCP SR packet generation mechanism (and in particular the NTP/RTP timestamps) is central to the skew detection mechanism proposed by the authors and can be severely impacted by such issues. Secondly, the selection of packet size can have a negative impact if it does not match the underlying sound system fragment size. Finally, OS scheduling jitter can introduce significant noise, as detailed in [8].

4. SKEW COMPENSATION

The exact symptoms of skew and thus the appropriate compensatory mechanism depend on the precise operation of the audio application. A simple application that initially buffers a fixed amount of data in a receiver before playout will see overfill or underfill depending on the difference in audio clock rates. In this case, a strategy similar to that of [4] that periodically inserts or deletes samples will help maintain the buffer at an optimum level rather than letting it move to either extreme.

More complex adaptive jitter buffer applications that utilise receiver system clock timestamps introduce further complications. For example, the hybrid playout algorithm developed by the authors utilises synchronised system clock time to precisely determine one-way delays on a per packet basis, but uses sender RTP timestamps in these calculations. Knowledge of the skew between sender audio and receiver system clocks can thus be used to avoid the gradual distortion of one-way delays that would otherwise occur. Although such distortion will affect the operation of adaptive playout algorithms such as those proposed by [9][10][11], the impact is more severe where actual end-to-end delays are required as in the case of the hybrid algorithm. Skew between sender and receiver audio clocks will in any event result in overflow or underfill of buffers and a mechanism similar to that outlined above is still required for effective playout.
This paper presents a high level mechanism that integrates the RTCP Sender Reports with NTP to accurately measure audio and system clock skews. Although the results are very encouraging, a number of implementation issues relating to Operating System (OS) and sound card design require further analysis. With regard to NTP performance, testbed results indicate that it can provide robust. Although not detailed in this paper, the issues relating to the design of a robust NTP subnet are well documented elsewhere [12].

Finally, to make use of this mechanism, it needs to be incorporated and tested within an actual VoIP application. Integration details are very much dependent on the operation of the application. As such, the authors hybrid playout algorithm for optimising VoIP quality is currently being extended to incorporate this work.

6. ACKNOWLEDGEMENTS

We would like to thank Richard Akester of UCL and Orion Hodson of ICIR, Berkeley for their help. Thanks also to Prof. Gerry Lyons of the Dept.of IT, National University of Ireland, Galway for his assistance.