

Characterization of Multimedia Streams of an H.323 Terminal

Hani ElGebaly, Emerging Products Development, Intel Architecture Labs, Intel Corporation

Index words: H.323, Conferencing, Internet, G.723.1, H.263

Abstract

A study of multimedia performance over the Internet requires accurate representation of the workload model. Measurements of multimedia conferencing applications provide a snapshot of real workloads. This paper presents a characterization for video and audio traffic transported over the Internet by multimedia conferencing applications following the H.323 set of standards defined by the International Telecommunication Union (ITU-T).

Our goal is to investigate multimedia traffic multiplexing issues at the host. We are not concerned with the Internet infrastructure or the store and forward technology deployed. The traces collected in this study are done at the multimedia source host. They provide information about packet length, inter-arrival time, jitter, overhead, and burstiness.

These traces help in understanding local-induced effects on the multimedia traffic mix. Many of these effects are primarily due to limitations of operating systems, bandwidth sharing, or the choice of traffic parameters. These local effects contribute significantly to traffic delay or the abuse of network bandwidth and congestion.

A few tradeoffs are involved in the choice of multimedia traffic parameters. For example, a tradeoff exists between the network protocol overhead and the local latency of multimedia packets. Another tradeoff exists between the video packet size and the burstiness of the video packets. In this paper, the various tradeoffs involved in generating audio and video packets are discussed. We focus on the audio and video codecs defined in Recommendations G.723.1 and H.263 of the ITU, respectively. Both codecs are extensively utilized by H.323 developers because of their efficiency, popularity, and suitability for transmission over low bandwidth pipes. We derive an optimal operating point for both audio and video traffic for better bandwidth efficiency and acceptable latency.

We also discuss the interaction of conferencing media components as they are multiplexed on the host to be

transmitted to a peer terminal. Lack of appropriate multiplexing algorithms can lead to one or more media components oversubscribing to the shared bandwidth and penalizing other participants. A new performance qualifier is introduced in this paper. This qualifier is the number of interleaved video bytes scheduled for transmission between audio packets. This number turned out to be a good local indicator for audio jitter and latency.

Introduction

Advances in computer technology, such as faster processors, faster modems, Intel MMX™ technology, and better data compression schemes have made it possible to integrate audio and video data into the computing environment. A new type of video conferencing is now possible: desktop video conferencing. Desktop video conferencing applications include telecommuting, corporate meetings to cut travel costs, family gatherings, Call Centers, etc. Banks, shopping centers, retail centers, and so on can be more efficient and cut a lot of overhead by providing video-conferencing customer service centers for customers to dial into.

One of the most prominent enabling technologies for desktop video conferencing is the H.323 standard developed by the ITU-T. Recommendation H.323.[6] describes terminals, equipment, and services for multimedia communication over networks such as the Internet.

This paper addresses issues of the multimedia traffic sources of an H.323 terminal. Issues such as packet format and multiplexing of audio and video frames at the host are studied. The traces collected in this study were done at the network edge. These traces provide information about the packet length, inter-arrival time, jitter, protocol overhead, and burstiness. Local-induced effects on the conferencing traffic can occur due to limitations of the operating system, bandwidth sharing, and lack of an accurate performance qualifier for choice of traffic parameters. The

collected traces help to understand these effects and eliminate them.

Measurement Methodology

This paper studies the packet format parameters and multiplexing issues of audio and video traffic at the transport layer before reaching the network. Traffic measurements capture packet overhead imposed by the Transport, Network, and Datalink layers' protocols. It also provides information on the local inter-arrival time, latency, and jitter.

We focused our study on audio and video codecs defined in the G.723.1 specification [3] and the H.263 specification [5], respectively. Although the mandatory codecs for the H.323 standard are G.711 [2] and H.261 [4], most of the H.323 terminals that connect to the Internet via modems through Internet service providers use G.723.1 and H.263 codecs as their preferred conferencing codecs. G.723.1 has lower bit rate (5.3/6.4 kb/s) than G.711 (64 kb/s). Hence, it is more suitable for transmission over low bandwidth links (e.g., 28.8 kb/s and 33.6 kb/s modems). However, for video conferencing, H.263 has better quality than H.261.

It is important to capture the worst-case behavior of the network such that the solutions proposed can have wide applicability. It is also important to observe the most congested time of the day over the Internet. We used a variety of comparable Internet service providers for establishing conferencing sessions.

Media Trace Format

Audio Format

G.723.1 [3] is a dual-rate speech-coding standard, which operates in low bit rate while maintaining high perceptual quality. It is recommended as the preferred speech codec for the ITU H.323 [6] conferencing standard over the Internet when the access link to the Internet has limited bandwidth (e.g., an ISP connection using a modem). The G.723.1 codec has two bit rates associated with it. These rates are 5.3 and 6.4 kb/s, the latter being of better quality. Both rates are a mandatory part of the encoder and decoder.

The collected measurements are applied to a traffic scheduler. The scheduler accepts audio and video packets and schedules them for transmission using the scheduling algorithm of interest. During the simulation run, measurements of packet sizes, inter-arrival time, latency, and jitter are computed. The scheduler is depicted in Figure 1.

The scheduler is a multithreaded application. Each traffic source has its own thread. The scheduler also has its own

separate thread. Additional traffic sources such as T.120 data or other video or audio codecs can be easily hooked to the scheduler.

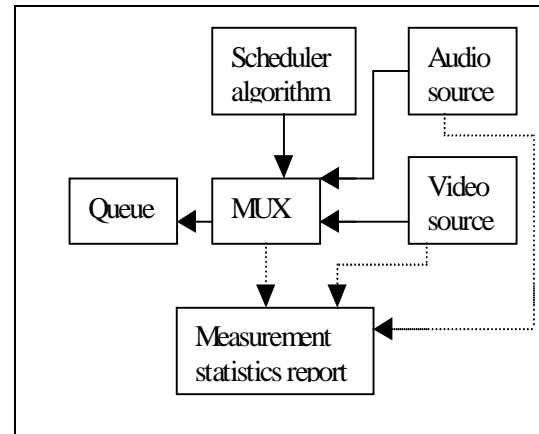


Figure 1: Architecture of the traffic scheduler

The G.723.1 encoder provides one frame of audio every 30 milliseconds. The audio frame size is 20 bytes of pulse-coded modulation (PCM) samples for the low rate (5.3 kb/s) and 24 bytes for the high rate (6.4 kb/s). Each application is set for a particular number of frames per audio packet before the conference starts. These values are negotiated when the call is established, and the least number prevails. Both applications are forced to use this number as the maximum number of frames incorporated in an audio packet.

Increasing the number of frames per audio packet improves the bandwidth utilization and decreases network packet overhead. However, it also introduces additional delay to the audio playback since a packet has to wait for all the audio frames to be accumulated before sending it across. This host-induced delay can be even more harmful especially with the timeliness requirement of real-time audio telephony.

Each audio packet has a Real-time Protocol (RTP) [7] header (12 bytes) that carries time-stamps, sequence numbers, a payload type, and a synchronization source identifier. In addition, a User Datagram Protocol (UDP) header is needed to carry UDP information about the packet (8 bytes). Then an Internet Protocol (IP) header of 20 bytes is also needed to carry routing information. Finally, since we are strictly constrained in bandwidth as we are expected to connect to the Internet via Internet service providers, an additional 5 bytes of point-to-point protocol (PPP) header is also required.

Video Format

The H.263 codec [5] is designed for a wide range of bit rates (10kb/s - 2 Mb/s). The codec supports five different resolutions. In addition to Common Intermediate Format

(CIF) and Quarter Common Intermediate Format (QCIF) that were supported by H.261, there is sub QCIF (SQCIF), four times CIF (4CIF), and sixteen times CIF (16CIF). SQCIF is approximately half the resolution of QCIF. 4CIF and 16CIF are four and sixteen times the resolution of CIF, respectively. Support for 4CIF and 16CIF allows the H.263 codec to compete with other higher bit-rate video coding standards such as the MPEG standards. H.263 is a hierarchical codec, i.e., some performance and error recovery options can be sacrificed for lower bit rate.

It is important to understand how the video source generates video data in order to analyze video-generated traffic. Video pictures are fragmented into multiple video fragments. Each fragment forms a video packet. Each packet is sent across the network after RTP, UDP, IP, and PPP headers have been added. Video is different from audio in the sense that the generated bit rate can be variable while the audio bit rate is usually constant. The ITU-T standard left this issue up to the application as to whether to use a fixed or variable bit rate for video [6]. Another important difference between video and audio media types is fragmentation. Video can be fragmented because a video-generated frame can be quite large; audio is not fragmented. Fragmentation means a video frame is divided into multiple fragments. Each fragment forms a video packet that is sent across the network. There is an upper limit on the maximum fragment size used by the fragmentation process.

A third major difference between audio and video is the burstiness of the video source due to fragmentation, which causes multiple fragments to cluster in a small interval of time. This burst of packets can lead to packet loss, latency, or at least some amount of unfavorable jitter.

In this section, we focus on the characterization of generated video traffic after fragmentation and the implied tradeoffs when enforcing different maximum fragment sizes for video packets.

Smaller fragment size results in shorter inter-arrival time yet bandwidth efficiency decreases because of larger packet overhead. However, larger fragment size results in longer inter-arrival time yet maintains good utilization of bandwidth. As the maximum fragment size limit increases, larger video packets will occur more often. These packets will require more time to be transmitted and hence the video latency increases.

An experiment was conducted to study the video packet sizes (number of bytes per packet) for different maximum fragment limits as generated by our video source. We set the maximum fragment size to four different values (128, 256, 512, and 750 bytes). We ran four experiments, each with a different maximum fragment limit and collected the

corresponding video packet sizes generated after the fragmentation module.

Figure 2 shows how video packet sizes are distributed for different maximum fragment sizes. The figure shows that by using a maximum fragment size of 750 bytes, 70% of the video packets had a size in the range of 50-250 bytes. Less than 10% of the total generated video packets were more than 512 bytes. For a fragment size of 512 bytes, 75% of the packets had sizes between 50-250 bytes. By decreasing the fragment size further to 256 bytes, more than 60% of the packets were in the range of 150-250 bytes. For a 128 maximum fragment size, the video packet sizes were equally distributed in the range of 50-100 bytes, and 100-150 bytes. Generally, the majority of the packets for all maximum fragment sizes were less than 256 bytes. This can mean that the maximum fragment size choice should be in the range of 256-512 bytes.

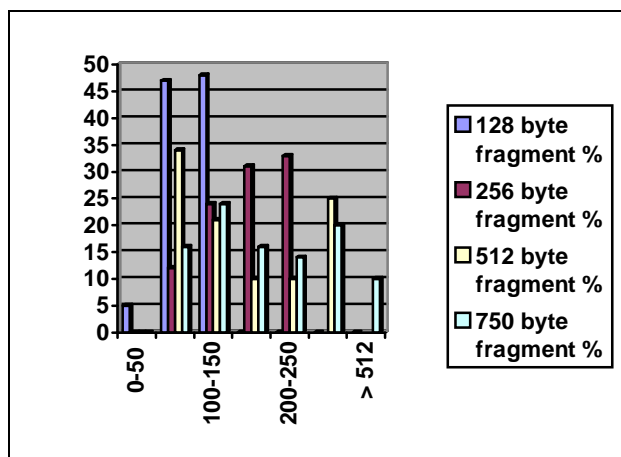


Figure 2: Byte distribution for video fragment sizes

Determining Traffic Parameters

Audio Traffic

It has been demonstrated that the choice of the number of frames per packet affects both local latency and protocol overhead (i.e., bandwidth utilization). We are primarily concerned with the latency induced by the audio packet preparation manager that buffers the captured audio frames and synthesizes them into audio packets. This latency is computed before the packet is actually sent on the network, and it only accounts for local capture and packet preparation delay. The choice of the number of frames per audio packet should minimize both local latency and packet overhead.

Results for Uncompressed Audio Packet Header

Figure 3 shows the percentage packet overhead for both the high and low bit rate G.723.1 audio codec. The packet

overhead decreases as the number of frames per packet increases. Intuitively, high bit-rate audio has slightly less overhead when compared to low bit-rate packets.

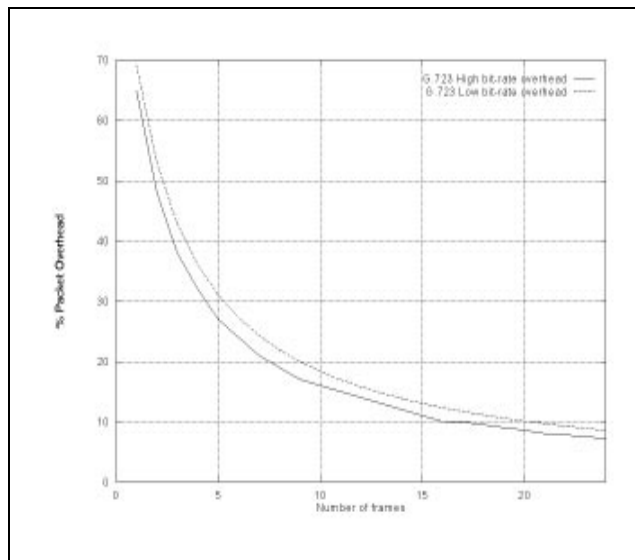


Figure 3: **Packet overhead for low and high G.723.1 audio**

In Figure 4, the number of audio frames per packet is plotted for both rates against normalized packet overhead and latency. The latency is normalized against the maximum latency value exhibited by the largest audio packet (24 audio frames per packet) used in our experiment. The packet overhead is normalized against the maximum overhead exhibited by an audio packet. An audio packet with only one audio frame has the maximum packet overhead as given in Figure 3. As the number of audio frames per packet increases, the packet overhead decreases. The overhead decrease is explained by more information bytes (audio frames) being included in a packet with a fixed header (RTP+UDP+PPP+IP).

The latency increases as the number of frames per packet increases. This is expected since more audio frames require more time to be captured and buffered. Hence, a tradeoff exists between packet overhead and local latency for audio packet transfer. The intersection of the latency and the overhead curves provides the choice of the number of frames that has the minimum latency and least overhead.

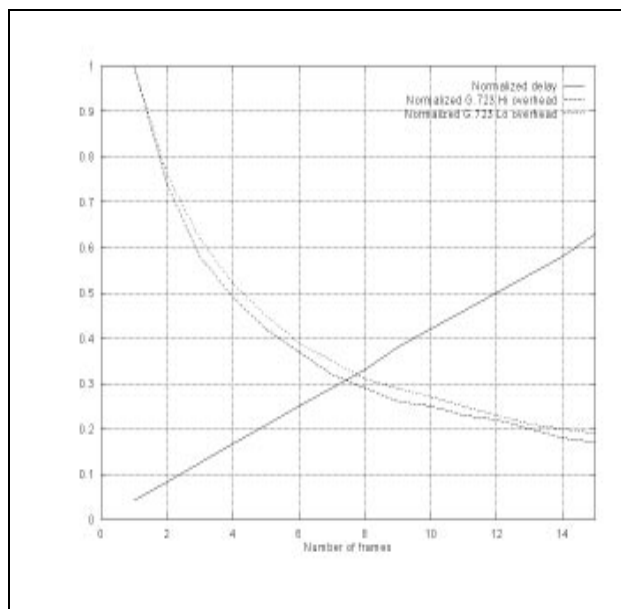


Figure 4: Packet overhead versus latency

The optimal point for the number of frames based on this plot is between seven and eight for both G.723.1 audio bit rates. This value should be used by terminals that target packet efficiency as well as low latency for audio packets.

Sometimes, the audio local latency is reduced at the expense of packet overhead, in order to achieve latencies acceptable to users. In fact, a few applications are willing to sacrifice some protocol overhead for achieving better audio latency. Some H.323 terminals use a value of 3 or 4 for the number of frames per audio packet in order to reduce this latency.

Results for Compressed Audio Packet Header

Protocol header compression has been an active research area for the past couple of years especially after the maturity of the protocols and standards that drive audio and video streaming over the Internet. Besides IP and UDP, researchers have also investigated RTP header compression. Jacobson and Casner [1] proposed an approach for compressing RTP, UDP, and IP headers to be used over low-speed serial connections to the Internet. Examples of these links include low speed (e.g., 28.8 kb/s) modem connections to the Internet through Internet service providers.

This approach seems ideally suited to better bandwidth utilization of the media streams since the protocol overhead is significantly reduced. A few companies are currently working on the deployment of this approach inside the network interface infrastructure in order to improve multimedia conferencing over the Internet for terminals connected via low speed links.

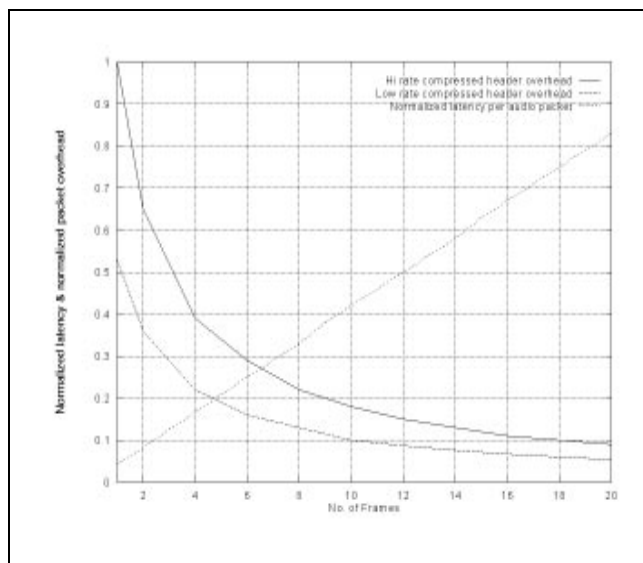


Figure 5: Packet overhead versus latency with compressed IP/UDP

Figure 5 shows a plot of the packet overhead versus latency after applying IP/UDP header compression. The curve suggests that the optimal number of audio frames is almost five for low bit-rate audio and six for high bit-rate audio.

Video Traffic

Another major decision in video source parameters is the choice of an appropriate maximum fragment size. The maximum fragment size contributes to the latency, network overhead, and the amount of generated traffic. As the size increases, larger video packets are generated. Large packets lead to increased latency and less protocol overhead. Reducing the maximum fragment size will lead directly to more packets being generated for the same number of video frames.

Video Packet Length Effect

Histograms of the video traffic are shown in figures 6-8. The choice of a maximum fragment size should capture most of the data clustering and at the same time should not penalize audio packets by generating large video packets that induce huge delays.

We chose 10 kb/s for the video bit rate, which is suitable for ISP-based Internet video conferencing terminals. We used the Quarter Common Intermediate Format (QCIF) for video streaming. Each trace lasted 15 minutes. The trace shows the video fragments generated from the H.263 codec after the RTP header is added. Measurements are taken before the addition of any UDP or IP packet header.

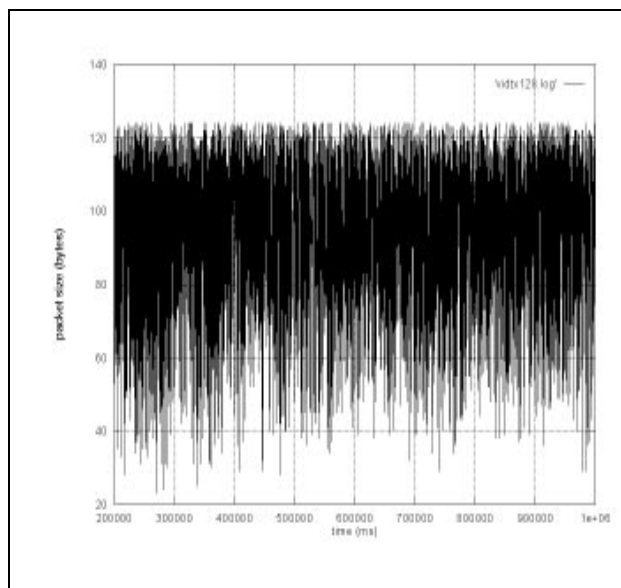


Figure 6: Video histogram for a maximum fragment size of 128 bytes

Figure 6 shows a plot of the video packet sizes against time with the maximum fragment size set to 128 bytes. The packets varied in size from as small as 25 bytes to the maximum value allowed, which is 128 bytes. The plot seemed too crowded as more packets are generated to make up for the small value of the maximum fragment size.

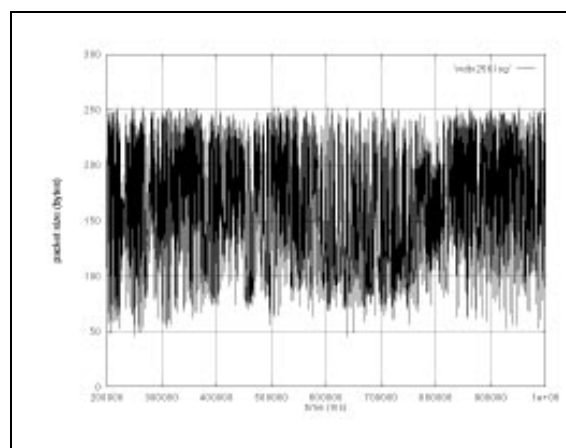


Figure 7: Video histogram for a maximum fragment size of 256 bytes

Figure 7 shows a plot of the video packet size against time where the maximum fragment size is set to 256 bytes. Packets varied in size from as high as 256 bytes to as low as 50 bytes. Figure 7 appears to be less crowded than Figure 6.

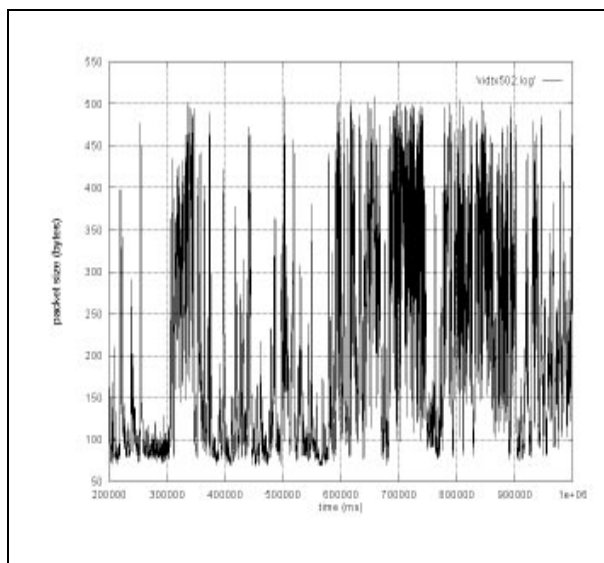


Figure 8: Video histogram for a maximum fragment size of 512 bytes

Figure 8 shows a plot of the video packet size against time with the maximum fragment size set to 512 bytes. The packet sizes ranged from 75 bytes to 512 bytes. The larger swing in the packet size range allowed for a less dense histogram compared to those in Figure 6 and Figure 7.

Table 1 shows the maximum fragment size and the corresponding mean, standard deviation, and number of generated fragments of the video traffic source.

Max Frag	Mean size	Standard Dev	Mean arrival	Frag no.
500	191.64	127.2	275.027	3680
256	169.38	51.358	224.40	4464
128	94.581	21.46	132.568	6382

Table 1: Mean and standard deviation for different fragment sizes

In Table 1, the smaller the maximum fragment size, the smaller the mean size of generated fragments and consequently the smaller the delay (smaller inter-arrival time). However, by examining the number of fragments generated for each maximum fragment size in Table 1, we note that as the maximum fragment size decreases, the number of generated fragments increases. This is expected, however, since the number of video frames to be fragmented and formed into packets is almost fixed for all maximum fragment sizes with which we experimented. In addition, as the maximum fragment size decreases, the packet overhead increases, and the bandwidth utilization decreases. Normally, it doesn't take any longer to send and receive fewer but larger video packets than it does to

send and receive more but smaller video packets, if the same amount of video data is played. In fact, the extra packet handling overhead may cause the reverse. However, larger video packets can cause audio latency as the bandwidth is shared, and this is the major concern.

Video Packet Inter-Arrival Time and Jitter Effect

Another important measure we can deduce from the data provided by these graphs is the mean inter-arrival time. This measure provides an estimate for video packet latency induced locally at the host. We compute the rate of change of the inter-arrival time, which is a measure of video packet jitter. As the maximum fragment size decreases, it is expected that the number of packets generated within the same interval of time will increase. The measurements for inter-arrival time and inter-arrival jitter for maximum fragment sizes of 128, 256, and 512 are shown in figures 9-11.

Figure 9 shows a plot of the inter-arrival time and jitter of video packets against time where the maximum fragment size is set to 128 bytes. Note that many packets are generated with variant inter-arrival time and hence the jitter between packets is significant. As the maximum fragment size is increased to 256 bytes (as in Figure 10), the jitter is less significant as a smaller number of packets with less delay variations is generated. Increasing the maximum fragment size further to 512 bytes decreases the jitter significantly as shown in Figure 11.

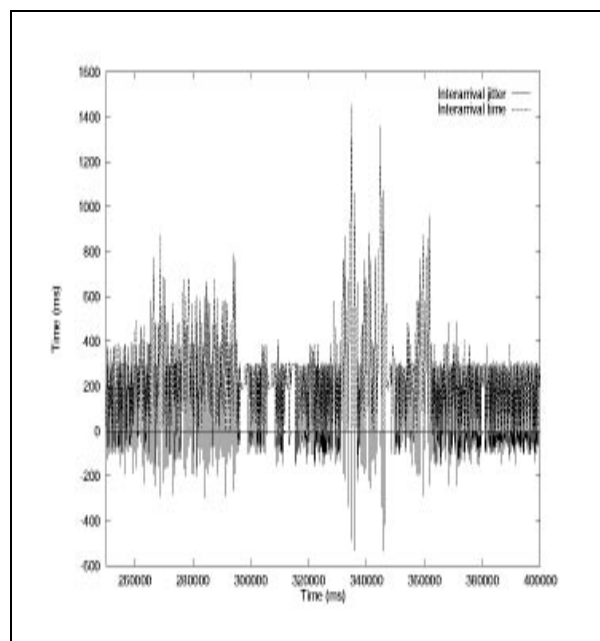


Figure 9: Inter-arrival time and jitter for video maximum fragment size of 128 bytes

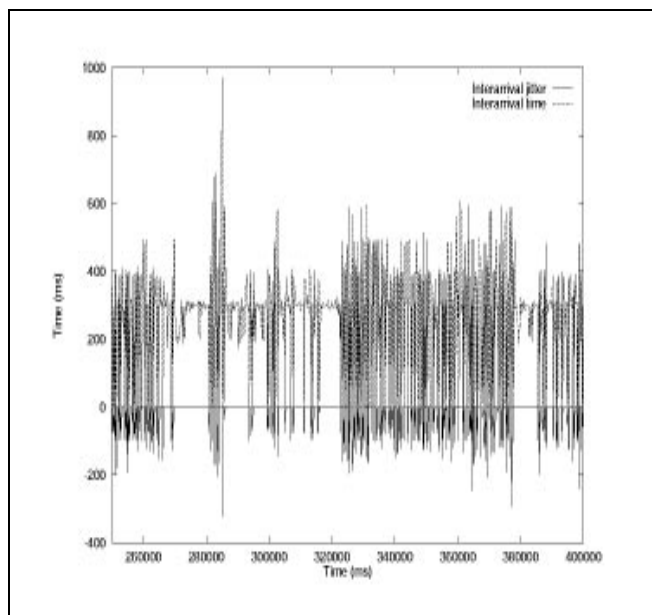


Figure 10: Inter-arrival time and jitter for video maximum fragment size of 256 bytes

In general, local jitter increases as the maximum fragment size decreases. This phenomenon is explained by the fact that more packets are generated for the smaller maximum fragment size during the same time interval. More packets with a variety of inter-arrival times result in a greater likelihood of inter-arrival time variation and hence, jitter.

Theoretically, as the maximum fragment size decreases, the sizes of the packets are normalized and jitter should be reduced. However, the increase in jitter in this case is also due to packet handling overhead in the transport stack from the increased number of packets.

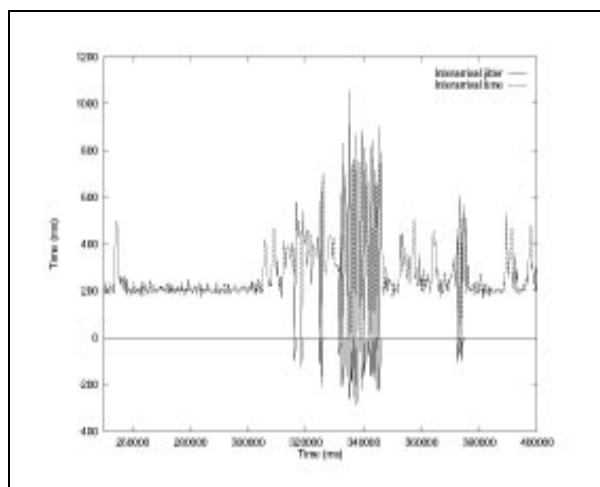


Figure 11: Inter-arrival time and jitter for video maximum fragment size of 500 bytes

Multiplexing Video and Audio Packets

Audio and video packets are generated by their respective sources and then funneled through a shared communication medium. Each media component should subscribe to a sufficient amount of bandwidth to meet quality standards. Lack of appropriate multiplexing algorithms can lead to one or more media components oversubscribing to the shared bandwidth and penalizing other participants.

A typical example of this multiplexing problem is when the scheduler dispatches all media types through a First Come First Served (FCFS) queue. All packets are serviced in the order they arrive without any attention being paid to their priority or timeliness. In this example, if the video data is bursty and a train of video packets is generated during an audio silent (inter-arrival) period, audio packets will be blocked for the duration of the video packet train. This duration may be sufficiently long to disturb the continuity of the audio speech and make it incomprehensible. To illustrate this example, we fed audio and video traffic to the scheduler shown in Figure 1. We chose a worst-case audio source that continuously generates audio packets of four frames each every 120 milliseconds. Video is of variable bit rate with maximum fragment size set to 256 and 512, respectively. We are interested in the number of video bytes that reside between audio packets during periodic silence intervals. If too many video bytes accumulate between two consecutive audio packets, severe delay (and jitter) may be experienced for the blocked audio packet.

We computed the mean number of video bytes interleaving audio packets from the plot. We depicted the host's extra inter-arrival time penalty to audio packets as time incurred by interleaving video bytes. Further, by knowing the maximum number of video interleaved bytes, we can show the maximum penalty incurred on the inter-arrival time of audio packets that will show at the receiver side due to the local multiplexing effect. We can also deduce the mean local audio jitter by computing the rate of change of the interlaced video packets over the experiment interval.

Figure 12 plots the number of video bytes residing between consecutive audio packets against time using an H.263 video source of maximum fragment size 256. Although it is commonly believed that using a smaller maximum video fragment size will lead to better audio performance, our experience proved otherwise. It is observed that the mean interleaved video bytes between audio packets for a video maximum fragment size of 256 is larger than the mean of the corresponding trace for a maximum fragment size of 512 as shown in Figure 13. Hence, reducing the video fragment size alone will not help the audio performance. On the contrary, it may lead

to irregular spacing in time between audio packets, which renders audio playback unintelligible.

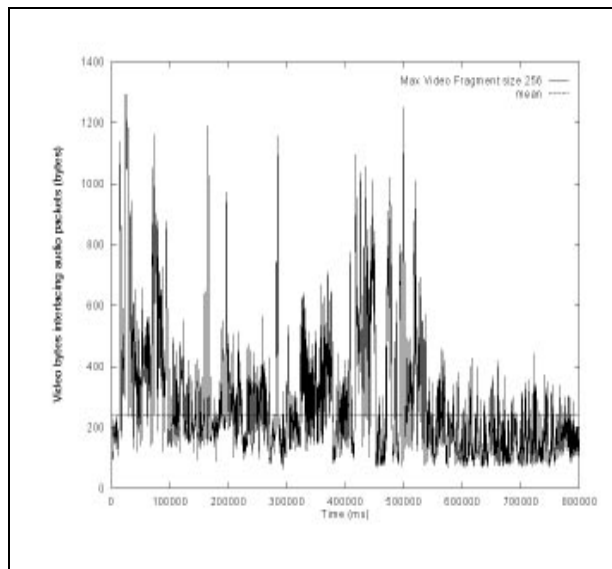


Figure 12: Interleaved video bytes between audio packets (maximum video fragment size 256)

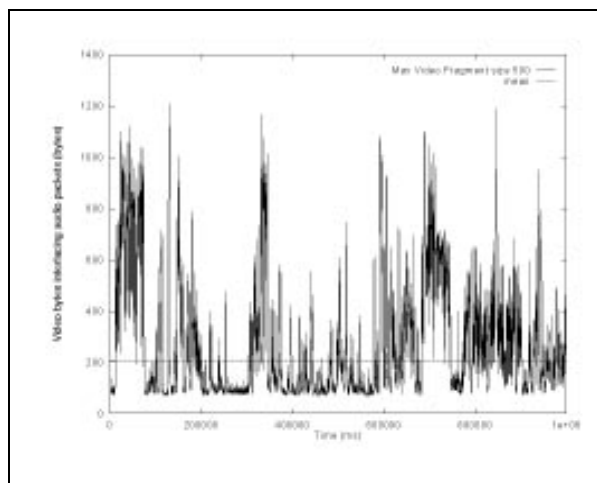


Figure 13: Interleaved video bytes between audio packets (maximum video fragment size 500)

The Problem with Media Multiplexing on the Host

Multiplexing audio and video packets over a shared bandwidth on the local host has its drawbacks. One major drawback we observed was where video oversubscribes its bandwidth. This leads to significant variable gaps between audio packets that causes audio inter-arrival delay and jitter.

There are two reasons for this problem. The first reason is directly related to the variable size of the video packet or

fragment. This can be controlled by fixing the size of the video packets, which can be inefficient. The second reason for the problem is related to the burstiness of the video source. Burstiness can be caused by the encoder, the packet fragmenter, or both. The remedy for the burstiness problem is a traffic regulator that controls the flow of video packets and maintains quality of service for all media types on the network. Other solutions include a recovery mechanism at the remote end for audio jitter effects, careful traffic monitoring and remote control of the violating traffic source, and use of priority schemes for audio at the network level and below. These solutions are beyond the scope of this paper.

Conclusion

Real video traffic workload is collected and analyzed in this paper. Worst-case G.723.1 audio traffic is used as the audio source input. The measurements collected during traffic analysis are packet length, inter-arrival time, jitter, and packet overhead.

Analysis of audio traffic revealed a tradeoff between the audio latency and the network bandwidth utilization. Increasing the number of frames decreases the packet overhead and increases the network utilization. However, increasing the number of frames per packet also increases audio local-induced latency by the time taken to buffer the frames before assembling them into packets. The optimal number of frames for minimum latency and minimum overhead is approximately seven per packet for an uncompressed header. This number will add latency overhead to the audio packet of 210 ms. Analysis of audio packets with compressed UDP and IP headers revealed the optimal number of frames to be five and six for low and high bit-rate audio, respectively. These numbers induce a local delay on the audio packet of 150 ms for low and 180 ms for high bit rates.

Analysis of the video traffic revealed that the mean packet size is less than 250 bytes for all fragment size limits. The standard deviation of packet size for all fragment sizes of choice is less than 180 bytes. The majority of the video packet sizes lay in the range of 50-250 bytes. This information suggests that a maximum fragment size of between 256 and 512 bytes is an appropriate choice.

It was also shown that as the maximum fragment size increases, the mean inter-arrival time between video packets increases, the number of generated fragments decreases, and the inter-arrival jitter decreases. In theory jitter should be reduced as the fragment size decreases. However, the increase in jitter in this case is also due to the packet-handling overhead (from the increased number of packets) in the transport stack by the packet assembler.

The local audio latency and jitter experienced by the audio-video scheduler are due to variations in video fragment size and burstiness of the video source. A new performance measurement was introduced. This measurement is the number of interleaved video bytes scheduled between audio packets. This number is a good indicator for audio latency and jitter at the host.

These results help assess H.323 terminal traffic locally at the host during a video conference. They provide useful information for tuning media traffic parameters. The results also provide a better understanding of the audio/video multiplexing problem over a shared bandwidth medium, and they enable the development of effective solutions.

Acknowledgments

The author would like to thank all his colleagues at Intel Architecture Labs. Special thanks and acknowledgment are extended to Steve Ing and Jose Puthenkulam of IAL for productive discussions during the various stages of this work.

References

- [1] Casner S., Jacobson V., "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links." Internet Draft, draft-ietf-avt-crtp-03.txt, July 1997.
- [2] ITU-T, Recommendation G.711 (1988)–Pulse Code Modulation (PCM) of Voice Frequencies.
- [3] ITU-T Recommendation G.723.1 (1996)–Dual Rate Speech Coders for Multimedia Communication Transmitting at 5.3 & 6.3 kb/s.
- [4] ITU-T, Recommendation H.261 (1993)–Video Codec for Audiovisual Services at p X 64 kb/s.
- [5] ITU-T, Recommendation H.263 (1996 –Video Coding for Low Bit-rate Communication.
- [6] ITU-T Recommendation H.323 (1996 –Terminal for Low Bit-rate Multimedia Communication over Non-Guaranteed Bandwidth Networks.
- [7] Schulzrinne H., Casner S., "RTP: A Transport Protocol for Real-Time Applications," draft-ietf-avt-rtp-04.txt, October 1993.

Author's Biography

Hani ElGebaly is the project technical lead for the H.32x protocols development within the conferencing products division at Intel Architecture Labs. He received an M.Sc. in Computer Science from the University of Saskatchewan, Canada, and a B.Sc. in Electrical Engineering from Cairo University, Egypt. Mr. ElGebaly is currently pursuing a Ph.D degree with the University of

Victoria, Canada. His primary areas of interest include multimedia conferencing protocols, embedded programming, fault tolerant systems, and computer architecture. His e-mail address is hani.elgebaly@intel.com