MediaPlayer™ versus RealPlayer™ - A Comparison of Network Turbulence

Mingzhe Li, Mark Claypool, Robert Kinicki
Computer Science Department
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609
{lmz|claypool|rek}@cs.wpi.edu

Abstract — The development of higher speed Internet connections and improvements in streaming media technology promise to increase the volume of streamed media over the Internet. The performance of currently available streaming media products will play an important role in the network impact of streaming media. However, there are few empirical studies that analyze the network traffic characteristics and Internet impact of current streaming media products. This paper presents analysis from an empirical study of the two dominant streaming multimedia products, RealNetworks RealPlayer™ and Microsoft MediaPlayer™. Utilizing two custom media player measurement tools, RealTracker and MediaTracker, we are able to gather application layer and network layer information about RealPlayer and MediaPlayer for the same media under the same network conditions. Our analysis shows that RealPlayer and MediaPlayer have distinctly different behavior characteristics. The packet sizes and rates generated by MediaPlayer are essentially CBR while the packet sizes and rates generated by RealPlayer are more varied. During initial delay buffering, MediaPlayer sends data at the same rate as during playout while RealPlayer can buffer at up to three times the playout rate. For high bandwidth clips, MediaPlayer sends frames that are larger than the network MTU, resulting in multiple IP fragments for each application level frame. From the application perspective, for low bandwidth clips, MediaPlayer has a lower frame rate than RealPlayer. Our work exposes some of the impact of streaming media on the network and provides valuable information for building more realistic streaming media simulations.

Index Terms — Empirical, MediaPlayer, RealPlayer, Streaming Multimedia

I. INTRODUCTION

The availability of high speed Internet access for home users and improvements in streaming media technology have led to an increase in the volume of streamed media over the Internet to the desktop. Increasingly, Web sites are offering streaming videos of news broadcasts, music, television and live sporting events. Users can watch these streaming video clips through a Web browser by simply clicking on a link and having the Web browser start up an associated video player.

Unlike typical Internet traffic, streaming video is sensitive to delay and jitter, but can tolerate some data loss. In addition, streaming video typically prefers a steady data rate rather than the bursty data rate often associated with window-based network protocols. For these reasons, streaming video applications often use UDP as a transport protocol rather than TCP, suggesting that video flows may not be TCP-friendly or, even worse, that video flows may be unresponsive to network congestion. A better understanding of the bandwidth and bandwidth patterns used by current streaming applications will help ascertain the threat of unresponsive traffic.

Research that attempts to deal with unresponsive traffic [CD01, FKSS01, MFW01, SSZ98] often models unresponsive flows as transmitting data at a constant packet size, constant packet rate or, as "firehose" applications, transmitting at an unyielding, maximum rate. Realistic modeling of streaming media at the network layer will facilitate more effective network techniques that handle unresponsive traffic flows.

The impact of currently available streaming media products will play an important role in the Internet. The use of commercial streaming products, such as the Windows Media Player™ (MediaPlayer) and RealNetworks RealPlayer™ (RealPlayer), has increased dramatically [JUP01]. Furthermore, it has been shown that commercial players typically use UDP as their transport protocol of choice [MH01, WCZ01], and therefore might be unresponsive. A better understanding of the network impact of commercial media products will produce better models of streaming flows and allow network architects to prepare for future Internet growth in streaming media.

In this work, we present an investigation of the size and shape of streaming flows, which we call *turbulence*[1], for both RealPlayer and MediaPlayer. We develop custom

[1] The term *footprint* is often used in systems work in the context of the basic size a piece of memory of some software. In a network, the size and distribution of packets over time is important, hence our word *turbulence*.

software, which we call *MediaTracker*, to play and record MediaPlayer video streams, and use it with previously developed software [WC02], called *RealTracker*, that plays and records RealPlayer video streams. We design experiments that simultaneously streams both RealPlayer and MediaPlayer videos that originated from the same content and the same Internet servers. We capture application level statistics and network level statistics and analyze the relationship between them and compare the two types of streams.

Our analysis shows that RealPlayer and MediaPlayer have distinctly different behaviors at both the network and application layers. At higher data rates, MediaPlayer streams have substantial IP fragmentation, while RealPlayer streams avoid fragmentation by transmitting smaller application-level frames. MediaPlayer streams have a constant bit rate (CBR), with uniform sized packets and packet interarrivals. RealPlayer, on the other hands, has more varied sized packets and packet interarrivals. During the initial buffering phase, RealPlayer streams have up to three times the steady playout rate, resulting in a burst of traffic for up to twenty seconds, while MediaPlayer streams at a steady playout rate for the entire clip.

The rest of this paper is organized as follows: Section 2 describes the experimental setup to study streaming traffic over the Internet; Section 3 analyzes the data obtained from our experiments; Section 4 briefly describes how results from Section 3 could be used to simulate streaming video; Section 5 describes some related work; and Section 5 summarizes our conclusions and presents possible future work.

II. EXPERIMENTS

A. Methodology

To carefully study the behavior of MediaPlayer and RealPlayer streaming video over the Internet, the following steps were taken as part of our measurement methodology:

- While both MediaPlayer and RealPlayer allow users to view performance statistics as clips play, neither allows the user to record the statistics. We built a customized version of MediaPlayer, called *MediaTracker*, to playback MediaPlayer clips and record statistics and used a previously developed customized version of RealPlayer, called *RealTracker* [WC02], to playback RealVideo clips and record statistics (See Section 2.B).
- In order to compare performance of RealPlayer and MediaPlayer, we accessed Web servers that had identical video content for both MediaPlayer and RealPlayer where the video servers themselves were co-located at the same or close to same server node (see Section 2.C).
- For each clip selected, we streamed identical MediaPlayer and RealPlayer clips simultaneously from the servers to one client concurrently receiving the video clips on the customized players. Both application level information and network packets statistics were recorded (see Section 2.D).
- We investigated network layer issues, such as packet arrival pattern and size, and application layer issues, such as video frame arrival patterns and frame rate, and the relationship among them (see Section 3).

B. Tools

We used several tools in our experiments, including MediaTracker, RealTracker, and a network packet sniffer tool – Ethereal for Windows 2000. Each of the tools is briefly described in the following sections.

1) MediaTracker

MediaTracker records application level information while playing back the MediaPlayer clips. Microsoft provides the Windows Media Software Development Kit (SDK)[2] for customized MediaPlayer development. The SDK supports two ways to build customized players: via an independent application or an ActiveX control object embedded in HTML Web page. MediaTracker uses an ActiveX object embedded in Java Scripts code.

The MediaPlayer ActiveX control is not included in the SDK, but comes with the free version of MediaPlayer. For alternate versions of MediaPlayer, the ActiveX control object ID varies and needs to be specified in the Web application. The ActiveX control for MediaPlayer 7.1 for Windows 2000 was used for our MediaTracker experiments. The latest version of MediaPlayer is 8.0, which only comes with Windows XP.

MediaTracker plays MediaPlayer clips using the core MediaPlayer engine just as MediaPlayer does; records the encoded bit rate, playback bandwidth, application level packets received, lost and recovered, frame rate, transport protocol, and reception quality; and supports a customized play list to automatic playback of multiple video clips.

MediaTracker saves all recorded information on the local disk. Although Java Script itself cannot write to the local hard drive, MediaTracker uses an ActiveX file system control to allow writing to the local disk after security verification via Internet Explorer.

---

[2] http://www.microsoft.com/windows/windowsmedia/create/develop.asp

2) RealTracker

RealTracker was originally developed in [WCZ01][3] for a Internet-wide RealVideo performance study. Unlike MediaTracker, which is a Web-based application, RealTracker is a stand-alone application. RealTracker was developed using RealNetworks' SDK[4] in Microsoft Visual C++ and uses the RealPlayer core video engine that come with the free basic version of RealPlayer. RealTracker was originally developed to use the core engine of RealPlayer version 8. However, we used RealTracker with RealNetworks' latest RealOne Player™.

RealTracker records statistics similar to that of MediaTracker, including encoded bit rate, playback bandwidth, frame rate, transport protocol. RealTracker also supports customized play lists for automatic playback of multiple video clips.

3) Ethereal

Ethereal[5] is a free network protocol analyzer for Unix and Windows. Ethereal captures data from a network and allows interactive browsing of the captured data. Ethereal includes a display filter language and the ability to view the reconstructed stream of a TCP session. We use Ethereal version 0.8.20 for Windows 2000 in our experiments, and captured all of the network traffic of streaming from the client to the video servers.

C. Clip Selection

To compare the behavior of MediaPlayer and RealPlayer under the same network conditions we selected servers that had both MediaPlayer and RealPlayer versions of the same videos. The goal was to make sure that the content of each clip set played in our experiments are identical for MediaPlayer format and RealPlayer format. That is, the MediaPlayer clips and RealPlayer clips should be encoded from the same media source and therefore have the same length and the same sense. Furthermore, we wanted the network environment to be the same for each clip set, so that the experimental results are comparable. To guarantee the above requirements, we:

- Selected clip sets from the same website which provides the same clip in both high and low encoded data rates and for both MediaPlayer and RealPlayer formats. The length of the clips should be between 30 seconds and 5 minutes.

[3] RealTracker was formerly known as *RealTracer*.

[4] http://www.realnetworks.com/resources/sdk/index.html

[5] http://www.ethereal.com/

Data Set		Encode (Kbps)	Clip Info.
1	R-h/M-h	284.0/323.1	Sports
	R-l/M-l	36.0/49.8	
2	R-h/M-h	268.0/307.2	Commercial
	R-l/M-l	84.0/102.3	0:39
3	R-h/M-h	284.0/307.2	Sports
	R-l/M-l	36.5/37.9	0:60
4	R-h/M-h	180.9/309.1	Music TV
	R-l/M-l	26.0/49.6	4:05
5	R-h/M-h	217.6/250.4	News
	R-l/M-l	22.0/39.0	1:47
6	R-v/M-v	636.9/731.3	Movie clip
	R-h/M-h	271.0/347.2	2:27
	R-l/M-l	38.5/102.3	

Table 1. Experiment data sets

- Selected clips in each set from the same subnet. Media clips that appear available from the same Web site may actually be served from a different subnet. Both MediaPlayer Servers and RealServers support redirection in that the IP address in the URL may not be the same one as the server that streams the video. Content distribution networks, like Akamai, provide a dynamic server address for each connection based on the client location and network status to reach the best playback performance. Thus, we confirm the network path of each clip by using Ethereal to locate the streaming server for each clip and using `tracert`[6] to discover network route. Only if the server addresses have the same network path and similar round-trip time did we use them as clips in our experiments.

- Selected one pair of high data rate clips (about 300 Kbps) and one pair of low data rate clips (about 56 Kbps) for both MediaPlayer and RealPlayer in order to compare player differences at different streaming rates,. For one server, we were able to find one pair of very high data rate clips (about 600 Kbps).

The above criteria greatly reduced the number of clips available. We collect six sets of clips for our experiments with a total of 26 clips with varied contents, lengths, encoding data rates, all encoded in both MediaPlayer

[6] Built in command in Windows 2000.

video and RealPlayer video formats. The clips chosen in experiment data sets are shown in Table 1. Sets 1 to 5 have both a high data rate pair and low data rate pair while the sixth set also includes a very high data rate pair.

D. Experiment Setup

The experimental setup is aimed to reduce the effects of the client, thereby concentrating on the effects of the video on the network. The PC is a Pentium-4 1.8 GHz processor, 512M RAM, AGP 32MB video card, PCI sound card, PCI 10M base Network Interface Card running Microsoft Windows 2000 professional. The software tools are MediaPlayer version 7.1, RealNetworks' RealOne Player, Ethereal version 0.8.20. The PC is connected to the WPI campus network[7], which is in turn connected to the Internet.

During pilot tests, it was verified that at no time during playout of any of the selected video clips were the CPU or memory overly taxed.

All the experiments were run Monday through Friday from 3:00 pm to 6:00 pm, EST, between March 29 and April 11, 2002.

Both MediaPlayer and RealPlayer can use either TCP or UDP as a transport protocol for streaming data. For all our experiments, we forced the players to use UDP as the transport protocol.

Before and after each run, `ping` and `tracert` were run to verify that the network status is had not dramatically changed, say from a route change, during the run.

III. ANALYSIS

A. Network Conditions

The basic conditions of the network during the experimental connections are estimated from the round-trip time and number of hops for each data pair. The cumulative density functions (CDF) of `ping` and `tracert` results are graphed in Figure 1 and Figure 2, respectively.

[7] http://www.wpi.edu/Admin/Netops/MRTG/

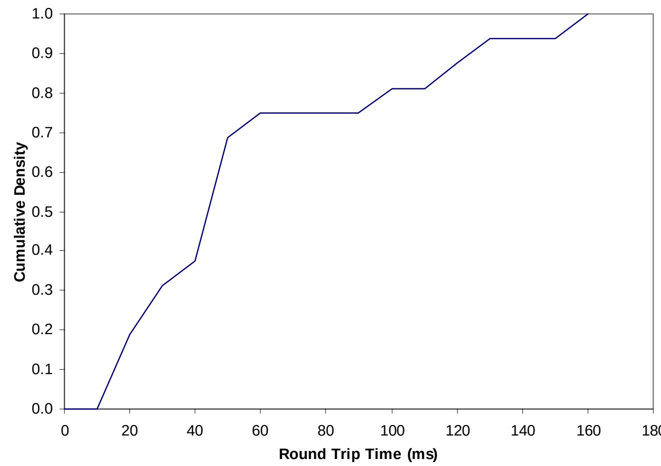


Figure 1. CDF of RTT

Figure 1 indicates that the experiments ran with a median round-trip time of 40 ms and a maximum round-trip time of 160 ms. Figure 2 indicates that most of the servers were between 15 and 20 hops away, results typical of other streaming experiments [LR01]. The average loss rate reported from `ping` was near 0%, again similar to results in [LR01], although we did observe a few packet losses during the experiments. We note that the goal of our study was not to compare the performance of RealPlayer and MediaPlayer under congested conditions, but rather to compare their performance under typical conditions.

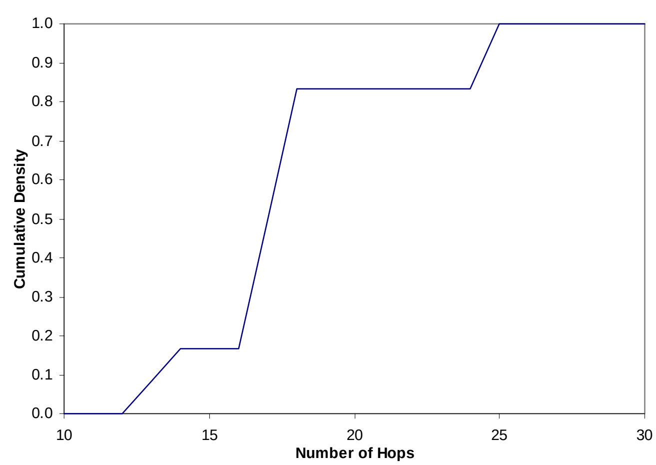


Figure 2. CDF of Number of Hops

B. Bandwidth and Encoding Data Rate

Table 1 shows the summarized clips information used in our experiments. The encoded data rate in the table is not from the link description provided by the Web page, but instead is captured by our customized video players. The bandwidth labels of the clips in the Web page typically indicate the connection bandwidth required to play the clip, but are sometimes different than the actually clip encoded rate. We observe that for the same advertised data rate, the RealPlayer clips always have a lower encoding rate than the corresponding MediaPlayer clip. For example, two clips are both advertised as needing a 300 Kbps connection, while the encoded rate for the RealPlayer clip is 284 Kbps and the encoded rate of MediaPlayer clip is 323 Kbps.

To further analyze the playback rate versus the encoded rate, Figure 3 plots the encoding data rate and average playback data rate points for each clip and graphs second order polynomial trend curves for both the RealPlayer clips and the Media Player clips. If the video clip were played back at the encoding rate, the trend curve would be the line y=x. In Figure 3, MediaPlayer tends to playback at the encoding rate, but RealPlayer plays out at a slightly higher average data rate than the encoded data rate. Thus, RealPlayer needs a higher average bandwidth than its encoding data rate for playback, hence, has an encoding rate at slightly less than the advertised value. MediaPlayer, on the other hand, needs only its encoded data rate for playback and so can encode at the advertised value.

From a content provider's point of view, the goal is to provide as good a quality as possible for the available bandwidth. Therefore, the lower encoding data rate required for RealPlayer suggests lower quality for the same bandwidth. However, RealPlayer's higher bandwidth consumption may be because of its buffering and playback mechanism, as described in Section 3.E.

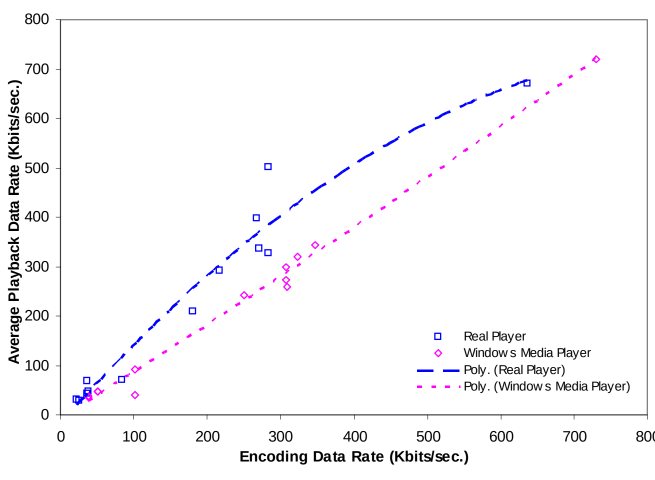


Figure 3. Average Playback Data Rate vs. Encoding Data Rate

C. IP Packet Fragmentation

Large application frames sent over UDP can result in IP fragmentation. Figure 4 shows the network layer packets' arriving pattern for one high encoding rate pair (a 250 Kbps MediaPlayer clip and a 217 Kbps RealPlayer clip). The MediaPlayer packets have a very regular pattern, with groups of packets and a constant number of packets in each group. Further investigation of the packet types using Ethereal reveals that each packet group is composed of one UDP packet and the remaining packets are IP fragments. All the packets in one group except the last IP fragment have the same size, which is 1514 bytes in our experiments. The size of the last fragment is different for each clip but is the same within each clip. Since the default Maximum Transfer Unit (MTU) for Windows is 1500 bytes[8], this suggests that MediaPlayer servers send large application layer frames that are then fragmented by the operating system to the size of the MTU.

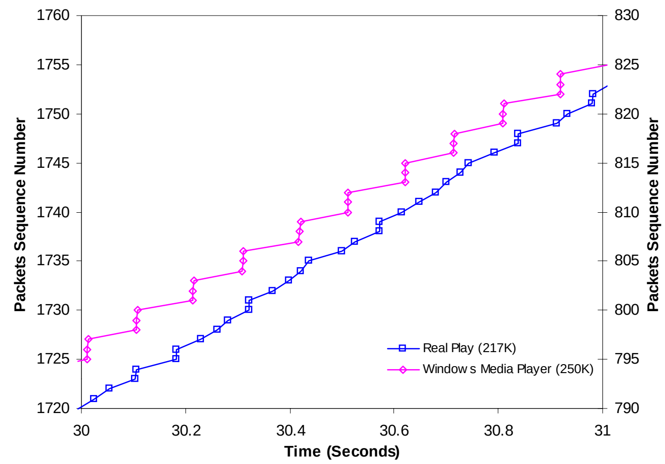
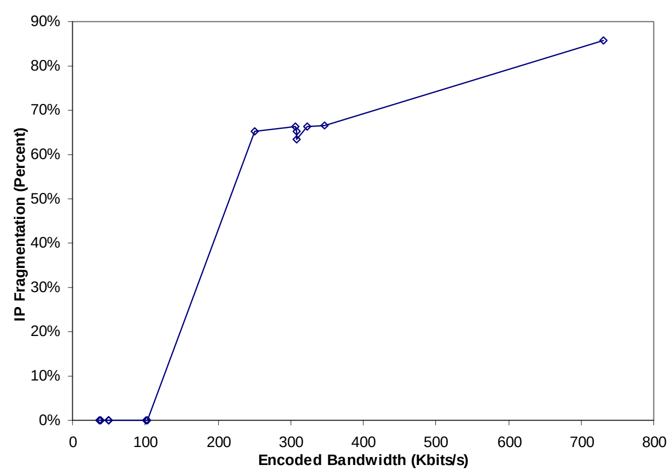


Figure 4. Packet Arrivals vs. Time

IP fragments were not observed in any of the RealPlayer traces which suggests that RealServers break application layer frames into packets that are smaller than the MTU, thus avoiding IP fragmentation.



[8] http://support.microsoft.com/default.aspx?scid=kb;EN-US;q140375

Figure 5. MediaPlayer IP Fragmentation and Encoded Data Rate

Figure 5 depicts MediaPlayer IP fragmentation observed for different encoding rates. The percentage of IP packets increases with an increase in the encoded rate. For example, 66% of packets are IP fragments for clips encoded at 300 Kbps, while there is no IP fragmentation for clips encoded at a rate below 100 Kbps.

IP fragmentation can seriously degrade network goodput during congestion, since a loss of a single fragment results in the larger application layer frame being discarded. Fragmentation at its worst can even lead to congestion collapse in the network [FF99]. Fragmentation based congestion collapse can occur when some of the cells or fragments of a network-layer packet are discarded (e.g. at the link layer), while the rest are delivered to the receiver, thus wasting bandwidth on a congested path.

D. Packet Sizes

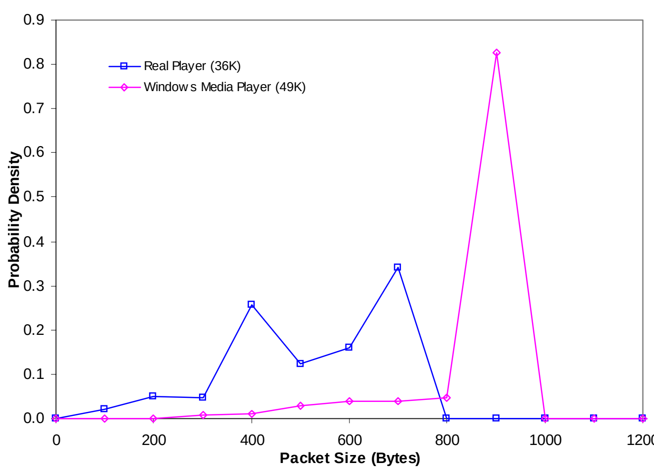


Figure 6. PDF of Packet Size for a Single Experiment (Data Set 1, Low Bandwidth)

The packet sizes for MediaPlayer traffic also shows stronger regularity than do the packet sizes in RealPlayer traffic. MediaPlayer packets have a high density at a particular size while the RealPlayer packet sizes are distributed over a larger range and do not have a single peak density point. Figure 6 depicts a Probability Density Function (PDF) of a typical run of one low bandwidth clip pair. Over 80% of MediaPlayer packets have a size between 800 Bytes and 1000 bytes. For high data rate clips, MediaPlayer has two high density distribution packet sizes, one at 1500 bytes contributed by the UDP and IP fragments, and another at the size of the last IP fragment, the remaining part of the large application layer packets.

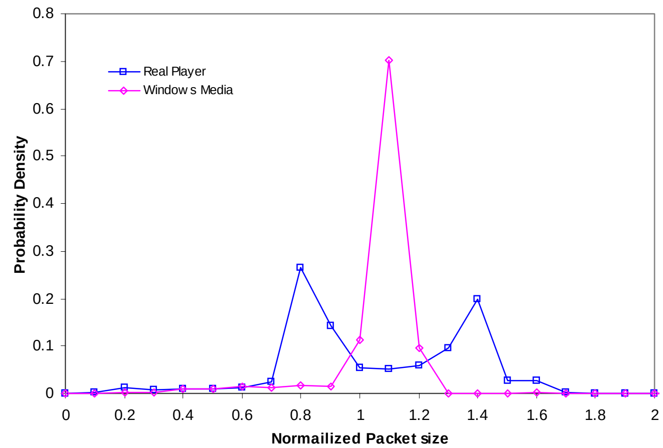


Figure 7. PDF of Normalized Packet Size (All Data Sets)

We summarize the packet size distributions for all experiments by normalizing the packets by the average packet size seen over the entire clip. Figure 7 shows a PDF of the normalized packets. The sizes of MediaPlayer packets are concentrated around the mean packet size, normalized to 1. The sizes of RealPlayer packets are spread more widely over a range from 0.6 to 1.8 of the mean normalized packet size.

E. Packet Interarrival Times

CBR traffic has fixed-size packets and a constant packet arrival rate. The difference in packet interarrival times, also known as jitter, can cause degradations to video perceptual quality that are as serious as packets loss [CT99]. Figure 8 shows a PDF of interarrival times of a typical run of one low data rate clip pair. MediaPlayer packets have approximately a constant time interval between packets, while RealPlayer packets have a much wider range of interarrival times.

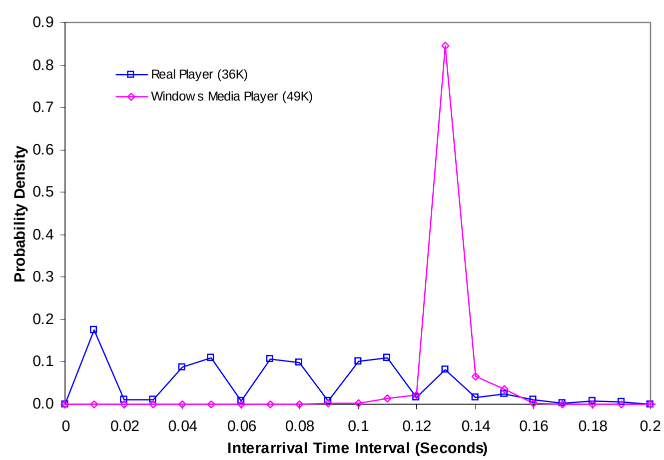


Figure 8. PDF of Packet Interarrival Times (Data Set 1, Low Bandwidth)

We summarize the packet interarrival distributions by normalizing the interarrival times to the average interarrival times for each clip. For high data rate MediaPlayer clips, we consider only the first UDP packet in each packet group to remove the noise caused by the IP fragments. Figure 9 shows Cumulative Density Functions (CDFs) of the normalized packet interarrival times. The CDF of packet interarrival times for RealPlayer has a gradual slope as packets arrive over all ranges of the normalized interarrival times. In contrast, the CDF of packet interarrival times for MediaPlayer is quite steep around a normalized interarrival time of 1, indicating that most packets arrive at constant time intervals.

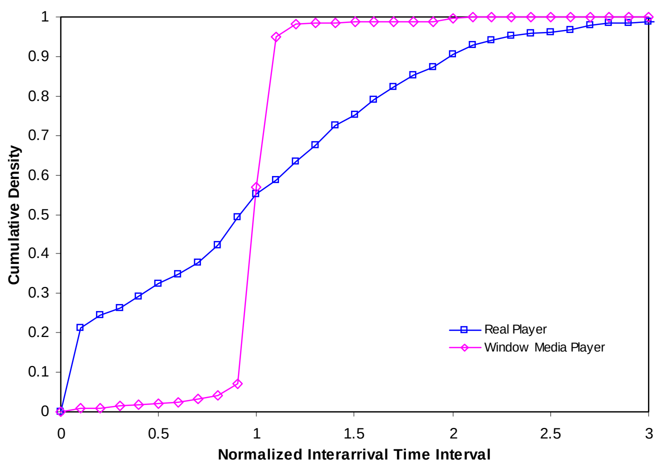


Figure 9. CDF of Normalized Packet Interarrival Times (All Data Sets)

This packet interarrival analysis combined with the packet size analysis from Section 3.D. suggests that MediaPlayer traffic is significantly more CBR than RealPlayer traffic.

F. Buffering Mechanism

Delay buffering is a well-known technique [RKTS94, SJ95] that streaming video players use for removing jitter. Data enters the buffer as it streams to the player, and leaves the buffer as the player displays the video. If network congestion causes a large interarrival time between packets, the player can keep the video smooth by playing buffered data. Both RealPlayer and MediaPlayer use delay buffering to remove the effects of jitter. Figure 10 depicts the bandwidth used over time for one data set. When the streaming begins, RealPlayer transmits at a higher data rate than the playout rate until the delay buffer is filled, at which time it transmits at the playback rate. The streaming duration is shorter for RealPlayer than for MediaPlayer since RealPlayer transmits more of the encoded clip during the buffering phase than does MediaPlayer. In contrast, MediaPlayer always buffers at the same rate as it plays back the clip, resulting in a less bursty data rate.

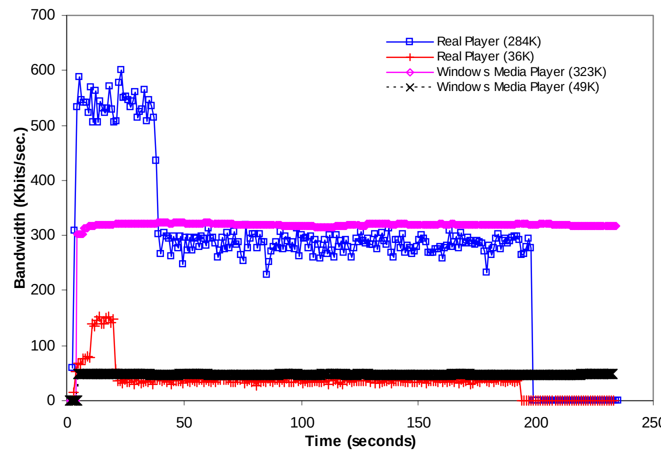


Figure 10. Bandwidth vs. Time for Single Clip Set (Data Set 1)

In Figure 10, the buffering rate of RealPlayer in proportion to the playout rate is higher for the low data rate clip than it is for the high data rate clip. Figure 11 depicts the ratio of the buffering rate to the playout rate for all RealPlayer clips (the ratio of buffering rate to playout rate for MediaPlayer clips is 1). The ratio of buffering rate to playout rate decreases as the encoding rate increases. For example, for the low data rate clips (less than 56 Kbps), the ratio of buffering rate to playout rate is as high as 3, while for the very high data rate clip (637 Kbps), the ratio of buffering rate to playout rate is close to 1, possibly because the bottleneck bandwidth is insufficiently small for a higher buffering rate.

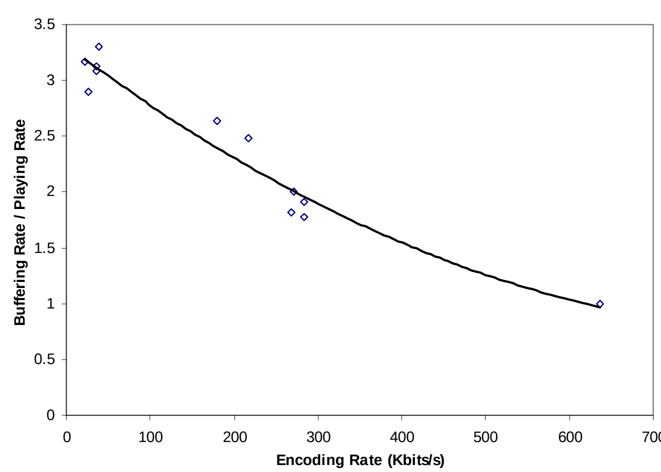


Figure 11. Buffering Rate/Playback Rate vs. Encoding Rate for RealPlayer Clips (All Data Sets)

If both RealPlayer and MediaPlayer have the same size buffer, RealPlayer will begin playback of the clip to the user before MediaPlayer. If RealPlayer and MediaPlayer begin playback of the clip to the user at the same time, MediaPlayer will have a smaller buffer and may therefore suffer from more quality degradations due to jitter. From the user point of view, RealPlayer either begins clip playback sooner or has a smoother playout than MediaPlayer. From the network point of view, RealPlayer generates burstier traffic that may be more difficult for the network to manage.

G. Packets Received by Network Layers

Packets received by the operating system may not be received by the application until some time later. MediaTracker allows us to record the time application layer packets are received. Figure 12 depicts the time the network layer receives the packets compared to the time the application layer receives the packets. The operating system receives packets in regular intervals of 100 ms, while the MediaPlayer application receives packets in groups of 10, once per second. Although Figure 12 only depicts the time over 4 seconds, this pattern occurs for all MediaPlayer clips over the entire clip duration. The delay between the operating system reporting receipt of the packets and the application reporting receipt of the packets may be caused by packet interleaving. Interleaving is a sender based media repair mechanism for multimedia applications. By dispersing the effects of packet loss crossing all interleaving packets, the receiver can mitigate the degradation of quality caused by packet loss without adding extra bandwidth [PHH98]. MediaPlayer interleaving may cause batches of packets to be reported as they are made available by the application layer. We are not able to gather application packets in RealTracker.

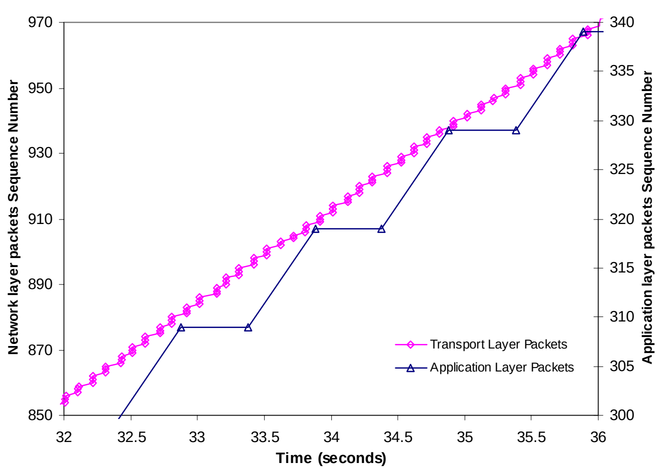


Figure 12. Packets Received by Network Layers for MediaPlayer for One Clip

H. Frame Rate

A basic unit of video performance is the rate at which frames are played. A higher frame rate yields smoother motion in a video. Figure 13 shows the results of frame rate versus time for a typical clip set. The two high data rate clips for MediaPlayer and RealPlayer both reach 25 frames per seconds, typically considered full-motion video frame rate. The lowest frame rate is for the low encoded MediaPlayer clip, which plays at 13 frames per second. The similarly encoded RealPlayer clip reaches a significantly higher frame rate than the MediaPlayer clip.

Figure 14 summarizes the frame rate results versus encoded data rate for all clip data sets. Each clip is plotted as a point based on its frame rate and encoded rate. For the low, high and very high clips, the average frame rate is plotted versus average encoded rate, along with standard error bars, and connected by lines. For low date rate encoded clips, MediaPlayer has a lower frame rate than RealPlayer, while for high and super high encoded data rate clips, MediaPlayer and RealPlayer playback at a similar frame rate.

Figure 15 summarizes the frame rate results versus playout bandwidth for all clip data sets. Each clip is plotted as a point based on its frame rate and average playout rate. For the low, high and very high clips, the average frame rate is plotted versus average playout rate, along with standard error bars, and connected by lines. Similar to the results for frame rate versus encoded rate, RealPlayer has a higher frame rate than MediaPlayer for the same bandwidth.

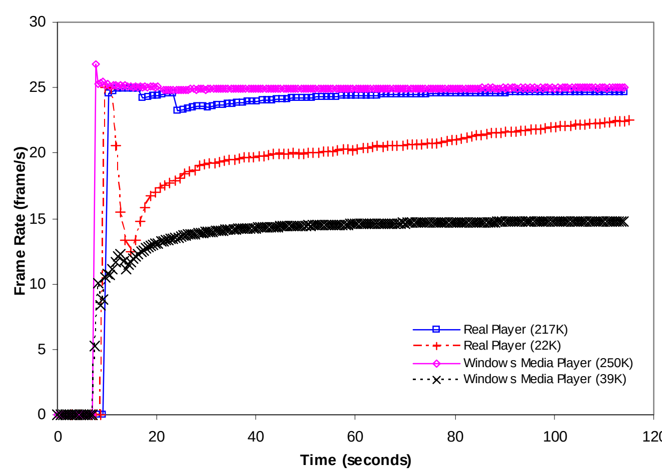


Figure 13. Frame Rate vs. Time for Single Clip Set (Data Set 5)

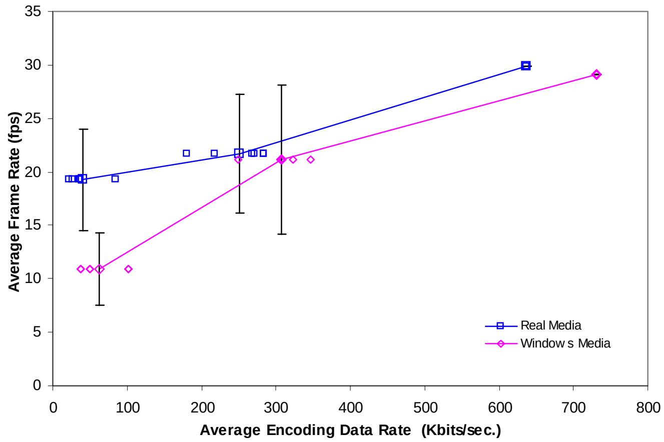


Figure 14. Frame Rate vs. Average Encoding Rate (All Data Sets)

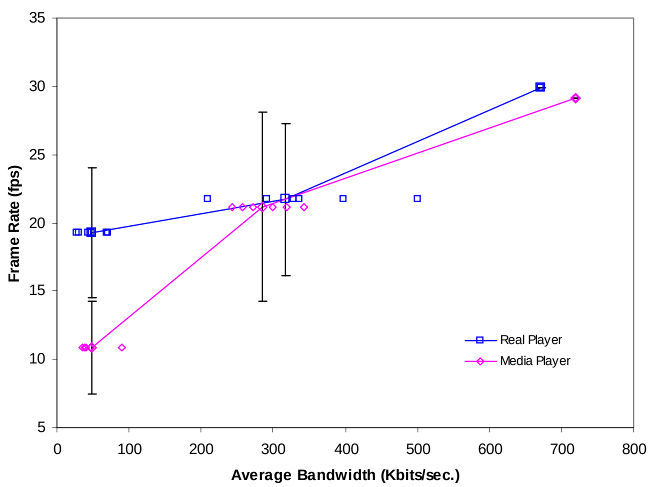


Figure 15. Frame Rate vs. Average Bandwidth (All Data Sets)

IV. SIMULATION OF VIDEO FLOWS

The data in section 3 provides insight into streaming video protocol designs, new network router queue management disciplines that react to streaming video flows, and interactions between streaming audio and traditional traffic. However, empirical experiments with live video streams are often difficult because of variable network conditions and the costs involved with deploying large numbers of video clients. Instead, simulations based on data from this paper can be an effective means of exploring network impact and enhancements of streaming video traffic. In the next paragraph, we briefly sketch out how we would build such simulations.

First, one could simulate either a RealMedia flow or a MediaPlayer flow. We cannot provide detailed network descriptions since we do not have extensive network information, but we can place the player in the simulated network by selecting an RTT based on Figure 1. Then, we would select an encoding rate and clip length from one of the data sets in Table 1. We would select packet sizes from distributions based on Figures 6 and 7 and generate packets at intervals based on distributions from Figures 8 and 9. MediaPlayer packets should include IP fragmentation rates based on Figure 5. RealPlayer data rates for the first 20 seconds (for low data rate clips) to 40 seconds (for high data rate clips) should be higher than the encoded rate based on Figure 11.

V. RELATED WORK

[MH00] presents the results of a brief study examining the traffic emanating from one popular Internet audio service using RealAudio. They found UDP to be the dominant download transport protocol, suggesting non-TCP congestion control. They observed consistent audio traffic packet sizes and rates that perhaps can be used for identifying flows or doing RealAudio simulations. We seek to build upon such work in measuring RealNetwork traffic by measuring RealVideo performance. Additionally, instead of measuring only network flow characteristics, we focus on more user-centric methods of performance evaluation.

[CWVL01] presents and analyzes a week long trace of RTSP packets from the University of Washington. They analyzed session length, session size and time of day correlations and the potential benefits from caching using their trace data and simulation. Instead of monitoring passing traffic from a sniffer, we analyze the performance from the client's perspective. This allows us to concentrate on system impact and performance of RealPlayer versus MediaPlayer rather than general RTSP-based multimedia traffic.

[MCCS00] describes the *mmdump* tool for parsing typical multimedia control protocols. Although the emphasis of their work is on presenting the tool itself, in demonstrating mmdump's utility they present results from monitoring live RTSP and H.323 traffic on AT&T's WorldNet IP network. Instead of traffic from one ISP, we provide playback analysis from six distinct servers and focus on video performance that both low-speed and high-speed clients would receive from RealPlayer and MediaPlayer.

VI. SUMMARY AND FUTURE WORK

The growing World Wide Web is increasing the volume of streaming video on the Internet. Commercial streaming video players, such as RealNetworks RealPlayer and

Microsoft Windows Media Player, promise to have a large influence on the impact of streaming video on the Internet. Previous empirical studies have focused on Internet traffic in general or have concentrated on a single commercial player.

In this work, we present an empirical study comparing the impact on the network for RealPlayer and MediaPlayer, the two most popular commercial video players in the world. To gather data for our study, we built a customized video player, called MediaTracker, which plays MediaPlayer clips and records performance statistics. We also used a previously developed video player called RealTracker that plays RealPlayer clips and records performance statistics. Using MediaTracker and RealTracker, we streamed clips selected from Web servers that offered the same content in both RealPlayer and MediaPlayer formats, and recorded network level statistics using a packet sniffer.

From analysis of the data, we find that high bandwidth MediaPlayer traffic can have up to 80% IP fragmentation rates, while RealPlayer has none. MediaPlayer packet sizes and inter-packet times are typical of a CBR flow, while RealPlayer packet sizes and inter-packet times vary considerably more. For all encoding data rate, RealPlayer buffers at a higher rate than does MediaPlayer, making RealPlayer burstier. For low encoding data rates, and the same average playout bandwidth, RealPlayer has a higher average frame rate than MediaPlayer.

The results obtained in this work are preliminary. Since we required content in both RealPlayer format and MediaPlayer format, we had a limited number of clips for study. Thus, although we chose a diverse set of clips from the servers, the range of possible video encodings is much larger. Despite this, the results presented here should be useful to network practitioners seeking insight into the practices and differences in commercial streaming video players. Network researchers should be able to use the results to produce more realistic video traffic for popular simulators, such as NS[9].

This work is only a brief beginning to the analysis of streaming multimedia traffic on the Internet leaving many areas for future work.

This study examined video clip traces obtained directly at a single player. It would be interesting to examine traces at an Internet boundary, such as the egress to our University, or at least at several players. Such analysis might reveal interactions between the media flows that our single client studies did not illustrate.

The use of TCP-Friendly congestion control is important for continued avoidance of Internet congestion collapse [FF99]. Both MediaPlayer and RealPlayer do have capabilities that employ media scaling to reduce application level data rates in the presence of reduced bandwidth. Studies similar to this one under bandwidth constrained conditions might help explore the feasibility of TCP-Friendliness (or, more likely the lack of TCP-Friendliness) in commercial media players.

ACKNOWLEDGEMENTS

We would like to thank Yubing Wang and Zheng Zuo for their development of RealTracker. We would also like to thank Yin Zhang for his preliminary version of MediaTracker.

VII. REFERENCES

[CD01] A. Clerget and W. Dabbous. "Tag-based Unified Fairness", In *Proceedings of IEEE INFOCOM,* Anchorage, Alaska, USA, Apr 2001.

[CT99] M. Claypool and J. Tanner. "The Effects of Jitter on the Perceptual Quality of Video", In *Proceedings of the ACM Multimedia Conference*, Vol. 2, Orlando, Florida, USA, Nov 1999.

[CWVL01] M. Chesire, A. Wolman, G. Voelker, and H. Levy. "Measurement and Analysis of a Streaming Media Workload", In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS)*, San Francisco, CA, USA, Mar 2001.

[FF99] Floyd, S., and Fall, K., "Promoting the Use of End-to-End Congestion Control in the Internet", *IEEE/ACM Transactions on Networking*, Aug 1999.

[FKSS01] W. Feng, D. Kandlur, D. Saha, and K. Shin. "Stochastic Fair Blue: A Queue Management Algorithm for Enforcing Fairness", In *Proceedings of IEEE INFOCOM*, Anchorage, Alaska, USA, Apr 2001.

[JRCC00] J. van der Merwe and R. Cceres and Y. Chu and C. Sreenan. "mmdump: A Tool for Monitoring Internet Multimedia Traffic", *ACM Computer Communication Review (CCR),* Vol. 30, No. 5, Oct 2000.

[JUP01] Jupiter Media Metrix. "Users of Media Player Applications Increased 33 Percent Since Last Year", *Press Release*, April 2001. http://www.jup.com/company/pressrelease-.jsp?doc=pr010403

[LR01] D. Loguinov and H. Radha. Measurement Study of Low-bitrate Internet Video Streaming, In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, San Francisco, California, USA, November 2001.

[MCCS00] J. van der Merwe, R. Cáceres, Y-H. Chu, C. Sreenan. "Mmdump - A Tool for Monitoring Internet Multimedia Traffic", *ACM Computer Communication Review*, 30(4), Oct 2000.

[MFW01] R. Mahajan, S. Floyd, and D. Wetherall. "Controlling High-Bandwidth Flows at the Congested Router", In *Proceedings of the 9th International Conference on Network Protocols (ICNP),* Nov 2001.

[MH00] A. Mena and J. Heidemann. "An Empirical Study of Real Audio Traffic", In *Proceedings of the IEEE Infocom*, pp. 101-110. Tel-Aviv, Israel. Mar 2000.

[PHH98] C. Perkins, O. Hodson and V. Hardman. "A Survey of Packet-Loss Recovery Techniques for Streaming Audio", *IEEE Network Magazine*, Sep/Oct, 1998.

[RKTS94] R. Ramjee, J. Kurose, D. Towsley and H. Schulzrinne. "Adaptive Playout Mechanisms for Packetized Audio Applications in Wide-Area

[9] http://www.isi.edu/nsnam/ns/

Networks", In *Proceedings of IEEE Computer and Communications Societies on Networking for Global Communication,* Toronto, Ontario, Canada, 1994.

[SJ95] D. Stone and K. Jeffay. "An Empirical Study of Delay Jitter Management Policies", *ACM Multimedia Systems,* Vol. 2, No. 6, pp. 267-279, Jan 1995.

[SSZ98] I. Stoica, S. Shenker, and H. Zhang. "Core-Stateless Fair Queuing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks", In *Proceedings of ACM SIGCOMM*, Vancouver, Canada, 1998.

[WC02] Y. Wang and M. Claypool. "RealTracer - Tools for Measuring the Performance of RealVideo on the Internet", *Technical Report WPI-CS-TR-02-12*, Computer Science Department, Worcester Polytechnic Institute, May 2002.

[WCZ01] Y. Wang, M. Claypool and Z. Zuo. "An Empirical Study of RealVideo Performance Across the Internet", In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, San Francisco, California, USA, Nov 2001.