Hierarchical analysis of RealMedia streaming traffic on an IEEE 802.11b wireless LAN

Tianbo Kuang and Carey Williamson

Department of Computer Science, University of Calgary

ABSTRACT

The popularity of multimedia streaming on the Internet, combined with the growing deployment of wireless access networks, augurs the converging usage of these two technologies in the not-too-distant future. Experience with wireless multimedia streaming on today's networks can provide valuable insights into the design of future wireless multimedia networks and applications.

In this paper, we present a measurement study of RealMedia streaming traffic on an indoor IEEE 802.11b wireless LAN. The traffic is analyzed hierarchically, from the application layer to the network layer to the data link layer. We focus on the traffic structure at each layer, and on interaction effects across layers.

Our main observation is that streaming quality is quite robust in all but the poorest channel conditions, despite the inherent burstiness of both the RealMedia application workload and wireless channel errors. Several factors contribute to these good results. First, although RealVideo is typically Variable-Bit-Rate (VBR) at the application layer, it is often streamed as Constant-Bit-Rate (CBR) at the network layer, reducing burstiness and thus the chances of packet losses due to buffer overflow in the network path. Second, while the wireless channel has bursty error characteristics, MAC-layer retransmission in 802.11b hides most errors from higher-layer protocols. Finally, the application layer's NACK-based error control is effective in recovering missing packets when needed. Our results demonstrate the viability of multimedia streaming on future wireless LANs.

Keywords: Multimedia Streaming, Wireless LANs, Network Traffic Measurement

1. INTRODUCTION

In recent years, there has been rapid growth in the use of two network technologies: Continuous-Media (CM) streaming, and broadband wireless networks. Multimedia streaming is widely used on today's Internet to provide real-time audio and video content to home, school, and office users with wired network connections. According to a study by Chesire *et al.*[1] in 2001, streaming media usage has increased significantly compared to several years ago (e.g., a 1999 study by Wolman *et al.*[2] showed that CM traffic accounted for 18-24% of Web-related traffic entering the University of Washington). Growth in the deployment and use of wireless access networks has been equally impressive (e.g., the ubiquitous installation of wireless LANs on university campuses).

The converging use of these two technologies is inevitable in the near future. Wireless networks free end users from the constraints of physical wires, enabling the retrieval of multimedia content from virtually anywhere at any time. However, wireless multimedia streaming is not trivial, due to the challenges of the wireless propagation environment (e.g., multi-path fading, channel interference, high error rates) and the stringent quality-of-service requirements of continuous media (e.g., delay, delay variation, and error resilience). Although compression can reduce multimedia file size dramatically, CM streaming over wireless networks still remains a challenge because of fluctuating channel quality during user sessions. Moreover, Variable-Bit-Rate (VBR) video exhibits burstiness[3–5] across many time scales, which may cause buffer overflows in a video delivery system.

Measurement of wireless multimedia streaming on today's networks can provide valuable insights into the design of future wireless multimedia networks and applications. In the literature, there is extensive research on wireless video coding, channel modeling, and error control, but no measurement studies of wireless video

Author contact information: Department of Computer Science, University of Calgary, 2500 University Drive NW, Calgary, AB, Canada T2N 1N4 Phone: (403) 220-6780 Fax: (403) 284-4707 Email: {*kuang, carey*}*@cpsc.ucalgary.ca*

streaming. The closest examples are Internet video streaming measurements on wired networks. For example, Mena *et al.*[6] studied the RealAudio traffic from a popular Internet audio server and found that RealAudio traffic shows behavior that is not TCP-friendly. They found that RealAudio packet traffic was bursty at small time scales, although the overall bit rate was constant at large time scales. Wang *et al.*[7] conducted a wide-area study of RealVideo traffic from several geographically-different servers to different users. They found that users generally achieve good quality video with an average frame rate of 10 frames per second (fps), though few achieve TV-quality full motion video (24 to 30 fps). Loguinov *et al.*[8] conducted an emulated study of streaming low-bit-rate MPEG-4 video to home users in more than 600 major U.S. cities. They reported results regarding packet loss, round-trip delay, one-way delay jitter, packet reordering, and path asymmetry. In another paper, Chesire *et al.*[1] analyzed the streaming media workload generated between clients at the University of Washington and servers outside.

In this paper, we focus on RealMedia[9] streaming from a wired-Internet RealMedia server to a wireless client. The measurements are conducted on an in-building IEEE 802.11b[10] wireless LAN (WLAN) in the Department of Computer Science at the University of Calgary. RealMedia is chosen because it is one of the most popular CM streaming applications on the Internet. The IEEE 802.11b WLAN is chosen because it represents a recent trend in broadband wireless network evolution. This paper is an extended version of our own prior work[11, 12] on this topic.

Our study demonstrates that video streaming on wireless LANs is viable, provided that the channel error rate is not too high and that appropriate error control mechanisms are used. The starting point for our study is an analysis of the traffic workload generated by the RealMedia streaming application (i.e., the output of the RealMedia codec). Such a workload study is useful in capacity planning for the design, deployment, and evolution of larger WLANs. This step is similar to a recent study by Fitzek *et al.*,[13] who analyzed a large number of MPEG-4 and H.263 compressed full-length video clips. They found that for Constant-Bit-Rate (CBR) video, there is no long-range-dependence (LRD) in most clips. For VBR video, their observations are consistent with the literature.[4, 5]

The second step in our study is a hierarchical analysis of RealMedia workloads. The goal is to understand the relationships between application-layer workload, network-layer workload, wireless channel characteristics (i.e., channel error rate, packet loss, retransmission, delay), and the networking protocols used (e.g., TCP, UDP, IEEE 802.11b). We explore these relationships by streaming RealMedia clips from a wired-Internet RealMedia server to a wireless client laptop at different (static) physical locations in our WLAN environment. The workload at the network layer is studied by capturing the traffic emanating from a RealMedia server using the `tcpdump` utility, which reflects all server effects (e.g., smoothing, shaping, and rate adaptation) on the traffic structure. The wireless channel characteristics are studied using a wireless "sniffer" to capture and analyze all WLAN traffic. The measurement results are used to explain the user-perceived quality of the streams. We believe this hierarchical approach is a useful method to study system performance and identify problems.

The rest of the paper is organized as follows. Section 2 provides some background on IEEE 802.11b WLAN technology, CM streaming, and the RealSystem streaming architecture. Section 3 describes the experimental setup for our measurement study. The measurement results at the application layer and the lower layers are presented in Section 4 and Section 5, respectively. Finally, Section 6 concludes the paper.

2. BACKGROUND

2.1. IEEE 802.11b wireless LAN technology

The IEEE 802.11b WLAN standard[10] is a high speed extension (currently up to 11 Mbps) of the original 2 Mbps standard[14] in the 2.4 GHz band. The standard specifies the physical layer and Medium Access Control (MAC) layer protocols used.

The physical layer allows four different data rates to be used for packet transmissions: 1 Mbps, 2 Mbps, 5.5 Mbps, and 11 Mbps. The data transmission rate can be adjusted on a frame-by-frame basis, to cope with wireless channel errors caused by loss, fading, and interference.[15] These random noise fluctuations affect the signal quality for the receiver, and can result in corrupted frames, particularly when sophisticated modulation

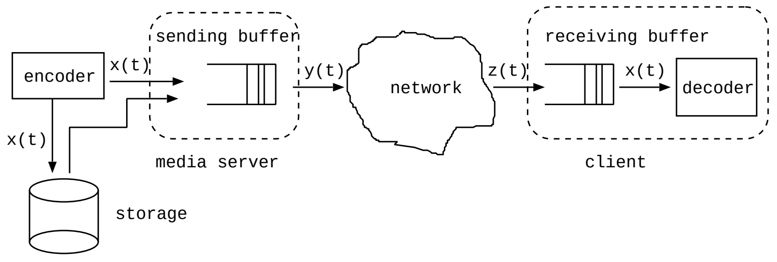


Figure 1. Logical View of Real-time Media Streaming

schemes are used for transmission. Data rate information is carried in the header of 802.11b frames (transmitted at 1 Mbps), so that the receiving station knows what clocking rate to use for the payload portion of the frame as it arrives.

The MAC layer regulates access to the shared (i.e., broadcast) channel in a wireless LAN. The typical MAC protocol used in 802.11b is Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA), also called Distributed Coordination Function (DCF). When a station wants to send a frame, it first senses the channel. If the channel is idle for a certain period of time (called the Distributed Inter-Frame Space (DIFS)), it transmits the frame. Otherwise, it waits until the channel becomes idle for another DIFS plus some random time. If the channel is still busy, the station doubles the random waiting period and repeats the process.

The 802.11b standard also defines error control mechanisms. To combat the unreliability of the wireless channel, the standard requires a receiver to acknowledge each correctly received MAC frame. Frames that fail the Cyclic Redundancy Check (CRC) at the receiver are simply discarded. If no acknowledgment is received shortly after transmission, the sender resends the frame, repeating this process as necessary until an ACK is received or until it reaches the maximum retransmission threshold (e.g., 4), at which point the sender gives up, leaving the problem to higher layer protocols.

The 802.11b WLAN can be operated in two different modes. In *infrastructure mode*, all stations communicate via an Access Point (AP) connected to a wired network. In *ad hoc* mode, there is no access point; stations communicate with each other in a peer-to-peer fashion. In this study, we use infrastructure mode, with the wireless client requesting media content from a server on a wired network. We did not use the Point Coordination Function (PCF) of 802.11b for multimedia, since this feature is not supported on our WLAN.

2.2. Continuous-media streaming

A logical view of a CM streaming system is shown in Figure 1. The encoder compresses the media data and writes it at rate $x(t)$ into a sending buffer (real-time coding) or onto a disk (offline coding). The media server then transmits the data at rate $y(t)$ to the client through a packet-switching network. Due to network effects, the media data arrives at the receiver with rate $z(t)$. The client plays back the data at rate $x(t)$, with a buffer to compensate for the mismatch between $z(t)$ and $x(t)$. The sending rate $y(t)$ has to meet two constraints: the receiving buffer must not overflow its finite capacity, and the receiving buffer must not be starved. CBR-encoded media data (i.e., constant $x(t)$) can be streamed in either CBR (constant $y(t)$) or VBR (time-varying $y(t)$) form. The same applies for VBR-encoded media, provided that the above constraints on $y(t)$ are met.

2.2.1. RealSystem media streaming architecture

RealSystem[9] is the Internet solution for audio and video streaming proposed by RealNetworks. They provide tools such as RealServer, RealPlayer, and RealProducer, corresponding to the media server, the client, and codec, respectively. The RealSystem architecture supports both real-time and on-demand streaming. We only study on-demand streaming in this paper.

The RealAudio and RealVideo contents are created in advance using RealProducer, and stored in RealMedia File Format (RMFF[16]) files. Before encoding, a target bit rate is chosen, based on the video quality desired for the intended audience. The bandwidth for the audio stream is allocated first, then that for the video stream.

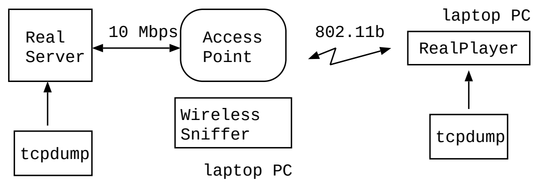


Figure 2. Experimental Setup for Measurements

One way that RealVideo achieves compression is by skipping frames when needed, so as to achieve a high frame rate for action scenes, and a low frame rate for low-activity scenes.

The RealVideo encoding can be CBR or VBR. While in VBR mode, bits from low motion scenes are "stolen" to make high motion scenes better quality while keeping the average target bit rate unchanged. A lot of video clips on the Internet are coded using *SureStream*[17] technology, wherein multiple versions of the same video with different rates are generated in advance and saved in the same file. In this approach, the client negotiates the suitably-encoded version of the stream to retrieve from the server.

The media data are stored as media packets. Each media packet in the encoded file has a stream number, timestamp, and size, plus a flag indicating whether it belongs to a key frame. The flag helps the server make decisions about which media packets to transmit, retransmit, or skip when problems occur. The RMFF file header contains the target rate, indicating how fast each stream should be delivered.

A streaming session is managed using the Real-Time Streaming Protocol (RTSP[18]). This control connection is established between the RealServer and the RealPlayer, usually using TCP. A separate connection is set up for the actual media data. A backchannel is also set up for the client to report receiver statistics or to request retransmssion of lost packets.[17] Besides the general start, stop, pause, and fast forward control functions, RTSP is also used to change the delivery parameters (e.g., delivery rate) of an ongoing streaming session (e.g., using `set_parameter`).

Before sending a media data packet, the RealServer encapsulates it using a *media packet protocol.* These protocols, such as the Real-time Transport Protocol (RTP[19]) or Real Data Transport (RDT[17]) protocol, facilitate the delivery and synchronization of real-time media data. The media packet is then carried by the transport-layer protocol. The default transport-layer protocol for audio/video streaming is UDP (User Datagram Protocol), though RealSystem supports TCP (Transmission Control Protocol) streaming as well (e.g., to traverse network firewalls). When a packet is lost, the client sends a negative acknowledgment (NACK) to ask for retransmission.

To overcome network delay and delay variation, the RealPlayer initially buffers incoming data for a few seconds before it starts playing back the streams. If network conditions change during the playback, the RealSystem uses Adaptive Stream Management (ASM[17]) to adjust the stream quality (e.g., to change the streaming rate, or to prioritize audio packets).

3. EXPERIMENTAL SETUP

3.1. Measurement facility

The experimental measurement setup uses three computers and one wireless access point (AP), as shown in Figure 2. The RealServer 8.0 software runs on a desktop Linux machine with a 1.8 GHz Pentium 4 CPU. One laptop acts as the RealPlayer 8.0 client, and another laptop runs the Sniffer Pro 4.6 wireless network analyzer software. Both laptops have an 800 MHz Pentium III processor and a Cisco Aironet 350 wireless network adapter. The AP is a Lucent RG-1000 residential gateway, connected to the server with a 10 Mbps Ethernet card. The AP uses infrastructure mode. The maximum retransmission limit at the MAC layer is 4.

3.2. Experimental design

3.2.1. Application-layer workload

For the application-layer traffic, we study on-demand RealMedia streaming packets stored in RMFF files, from which the coding bit rate, the traffic patterns, and frame-level statistics can be determined.

Four compressed RMFF files with different levels of scene changes and activity were examined. They are a *Seminar* talk with relatively little motion, a *TV Program* with a rich combination of activity levels and scene changes, a *Movie* clip, and a *Rock Concert* clip. The *Seminar* clip was compressed with multiple target rates using the *SureStream* technology, while the other three clips were all compressed with a single target rate.

Table 1 summarizes the clip information read from the file headers. The clips range in duration from 1 minute to 2 hours. Only one target rate is shown for the *Seminar* clip.

Table 1. Summary Information for the RealMedia Clips Used

	Rock Concert		*TV Program*		*Movie*		*Seminar*	
Item	Audio	Video	Audio	Video	Audio	Video	Audio	Video
Total Packets	432	2,850	14,640	44,577	4,704	101,893	26,724	141,868
Minimum Packet Size (bytes)	320	547	640	842	640	794	304	524
Maximum Packet Size (bytes)	320	685	640	1,009	640	1,007	304	708
Target Rate (kbps)	16.1	184	44.1	180.9	44.1	180.9	8.5	71.5
Duration (minutes)	1.1		28.3		48		128	
Target Frame Rate (frames/sec)	15		15		25		15	

3.2.2. Media streaming experiments

In the experiments, the RealMedia workloads are sent from the server to the client using UDP as the transport protocol. We study structural changes in the application workload, and how link-layer effects (e.g., wireless channel characteristics, MAC-layer retransmission) influence the higher-layer behavior.

In order to do this, we collect traces from the wireless client at different (static) physical locations, to represent different channel conditions. For ease of reference, we classify the wireless channel quality into four qualitative categories: *Poor*, *Fair*, *Good*, and *Excellent*. Table 2 summarizes these categories. These signal-strength categories are based on the Link Status Meter on the Cisco Aironet 350 devices. Prior work by Eckhardt and Steenkiste[15] shows how error rate is related to signal strength.

Table 2 also shows the average throughput and its standard deviation under different channel conditions. The throughput was tested with `netperf`[20] between client and server (test type: TCP-STREAM, receive socket buffer: 84 KB, duration: 10 seconds). Three observations are evident. First, the weaker the signal strength, the lower the throughput. Second, when the channel is *Excellent* the variability in the measured throughput is low. The same observation actually applies for a *Poor* channel. However, for *Fair* or *Good* channels, the variability is high, because of complicated fading characteristics. Third, we note that the maximum observed throughput of 4.6 Mbps for an *Excellent* channel is well below the nominal rate of 11 Mbps, reflecting the overhead of the IEEE 802.11b protocol. This result indicates that the 10 Mbps Ethernet connection between the server and the AP is not a bottleneck in our experiments.

The wireless channel characteristics are studied using the Sniffer Pro 4.6 wireless analyzer. The traffic characteristics at the network layer were studied using the `tcpdump` utility. For each streaming experiment, three traces are captured: one by `tcpdump` at the server, one by `tcpdump` at the client, and one by Sniffer Pro 4.6 close to the AP. Information such as packet size, type, and timestamp can be determined from these traces. Since a UDP packet usually conveys a single media packet, we can easily study interactions between layers.

Note that in the paper, two kinds of timestamps are mentioned: one is the (application-layer) *media timestamp*, the other is the (network-layer) *captured timestamp*. The media timestamp refers to the timeline information in the media packets for synchronization purposes. For example, a video packet with timestamp 100 is

Table 2. Qualitative Characterization of Wireless Channel

State	Signal Strength	Average Throughput (Mbps)	Std
Excellent	$> 75\%$	4.60	0.02
Good	45%-75%	3.76	0.26
Fair	20%-45%	2.14	0.28
Poor	$< 20\%$	0.09	0.15

coordinated with an audio packet with timestamp 101. These media timestamps are the same every time the stream is downloaded, since they are recorded in the RMFF file. The second (captured) timestamp refers to the time at which a network packet is seen by `tcpdump`. These timestamps are different every time the stream is downloaded. In the rest of the paper, we always refer to the first type of timestamp as the media timestamp, and the second one simply as the timestamp.

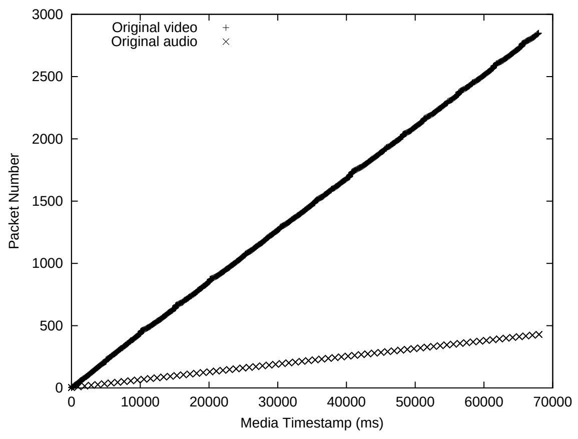
4. APPLICATION-LAYER WORKLOAD CHARACTERISTICS

In this section, we study the application-layer traffic characteristics of the RealMedia streams. We begin with characteristics at the media-packet level, followed by frame-level statistics. Finally, we briefly check the LRD of traffic at this layer.

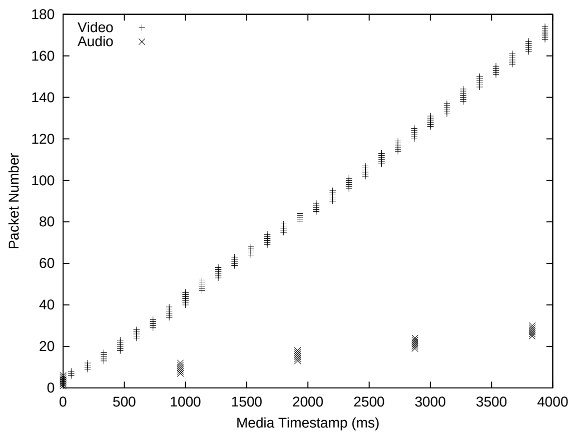
4.1. Media-packet-level characteristics

4.1.1. RealAudio traffic

Figure 3 plots the media packet number versus media timestamp for the *Rock Concert* clip. In each of the two graphs, the upper line represents video packets while the lower line represents audio packets.



(a) Entire Clip (68 sec)



(b) Initial Portion of Clip (4 sec)

Figure 3. Media Packet Number versus Media Timestamp (*Rock Concert*)

Figure 3(a) shows that the number of audio packets per second is roughly constant for the entire trace. However, Figure 3(b), a zoomed-in version of Figure 3(a), shows that audio packets are not evenly distributed in time. Instead, they appear as clusters of 6 packets with the same media timestamp, with time gaps in between clusters. This pattern indicates that the RealAudio codec generates audio traffic in “frames”, a basic synchronization unit for audio and video. All packets with the same media timestamp belong to the same frame.

A histogram analysis (not shown here) of the inter-packet times of the *Rock Concert* clip shows that two peaks are present: one at 0 ms, and the other at 958 ms. The peak at 0 ms represents packets that are part

Table 3. Frame-Level Statistics for Media Clips

Item	*Rock Concert*	*TV Program*	*Movie*	*Seminar*
Total Frames	511	18,230	9,661	77,442
Minimum Frame Size (bytes)	895	54	117	56
Median Frame Size (bytes)	2,930	1,684	1,041	864
Maximum Frame Size (bytes)	13,573	21,388	15,218	13,875
Mean Frame Size (bytes)	3,053	2,060	1,269	875
Standard Deviation (bytes)	1,449	1,915	849	753

of the same frame, as stated previously. The peak at 958 ms provides additional information about the traffic structure. Since there are 6 packets in an audio frame, each 320 bytes in size, the average data rate is

$$\frac{\text{6 pkts/frame x 320 bytes/pkt x 8 bits/byte}}{\text{0.958 sec/frame}} = \text{16.03 kbps (kilobits/sec)}$$

This calculation is consistent with the audio data rate shown in Table 1.

Similar observations apply for the other clips, except that the audio inter-frame time period is 1,856 ms for the *TV Program* and *Movie* clips. The *Seminar* clip has multiple inter-frame times since it was compressed with multiple target rates. However, a dominant inter-frame time is seen for each specific streaming rate studied.

4.1.2. RealVideo traffic

The RealVideo packets in Figure 3 exhibit the same clustered structure as the audio packets, though with three noticeable differences. First, the encoded media packet rate is different, as reflected by the differing slopes for audio and video streams in the plots. Second, the video stream uses variable-size packets and a variable number of packets per frame. Third, the inter-frame timing structure in the video stream is more subtle. Although the histogram of inter-packet times for each clip (not shown here) has two dominant peaks, there are several others. The peak at 0 ms again reflects packets belonging to the same frame. The other values, however, indicate that a RealVideo codec generates a variable-frame-rate video stream.

4.1.3. Implications of audio and video multiplexing

The multiplexing of RealVideo and RealAudio with the above patterns provides an explanation for the network-layer observations by Mena *et al.*,[6] who found that RealAudio traffic has a consistent bit rate at medium time scales and bursty on-off patterns at small time scales. This is because after sending a constant number of audio packets back-to-back, the encoder stops sending audio packets and starts sending video packets. When this process repeats periodically, the audio traffic exhibits an on-off pattern.

The foregoing explanation also suggests that the server may send video packets in a similar fashion at the network layer: trains of video packets, separated by audio packets. This issue is investigated further in Section 5.

4.2. Frame-level statistics

Frame sizes were calculated by summing all packets with the same media timestamp. A statistical summary of frame size characteristics appears in Table 3. All clips have a mean frame size that is larger than the median frame size, suggesting an upper tail to the distribution. This "heavy tail" property is most pronounced for the *TV Program* clip. The distribution of frame sizes is of limited interest here, however, since the inter-frame time varies. In particular, RealVideo occasionally skips frames so as not to exceed the target bit rate when the frame sizes are large.

A simple study of frame size versus frame index may exaggerate the burstiness of the traffic, if the inter-frame time is not considered. A rate trace avoids this problem, by dividing the frame size by its inter-frame time span. Moreover, to account for the "cluster" of frames that the codec handles, we add together the sizes of the frames within a cluster, and divide this sum by the time span between clusters. With a slight abuse of MPEG video compression terms, we call this a *Group of Pictures (GOP)* interval. Figure 4 shows the traffic profile

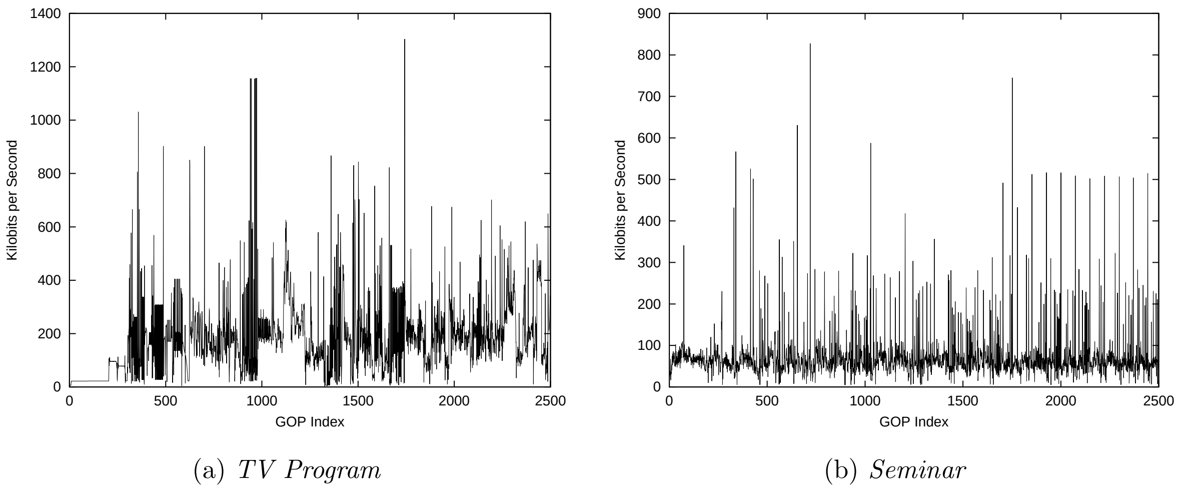


Figure 4. Application-Layer (Codec) Traffic Profile for Two Media Clips

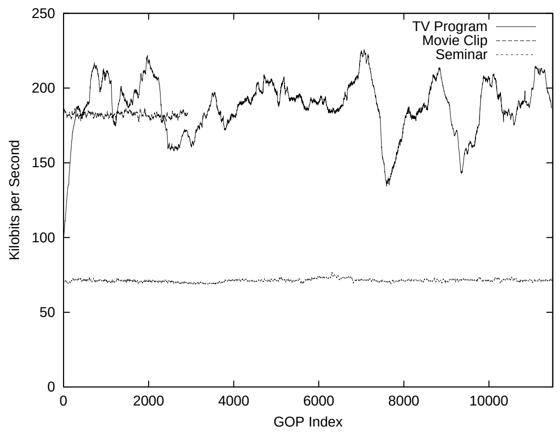


Figure 5. Smoothed Codec Traffic Rate for Selected Media Clips

calculated in this way for the *TV Program* and the *Seminar*. They both have some bursty peaks. However, the *TV Program* seems to have some slowly varying components (i.e., scenes) as well.

To better see the slowly varying components, a moving average filter is applied to each rate trace. The smoothing window size is 500 for all clips except *Rock Concert*, which uses a window size of 100 (since it has fewer data points). The results are shown in Figure 5. The *Movie* and *Seminar* clips are roughly CBR, while the *TV Program* still shows considerable variation, suggesting a VBR-coded stream. (The smoothed plot of the *Rock Concert* clip is not shown here because it is too short. When plotted on its own, its bit rate is CBR too.)

4.3. Long-range dependence

The VBR video traffic raises the issue of long-range dependence (LRD), which is briefly discussed in this section. Detailed analysis of the *TV Program* clip shows that this VBR-coded stream exhibits LRD, while some CBR-encoded streams (e.g., *Rock Concert*, *Seminar*) do not have LRD, and some CBR-encoded streams (e.g., *Movie*) may have weak LRD components. These results are consistent with the findings of Fitzek,[13] who reports that most CBR-coded variable frame rate H.263 streams do not exhibit LRD. However, they also found a few CBR traces that have weak LRD.

One possible explanation is that the LRD primarily represents the intrinsic nature of video. If an encoder is running without constraint (i.e., to generate a pure VBR trace), its output will reflect the underlying characteristics of the video. However, for CBR video, the encoder is forced to discard some bits, even frames, thus breaking the underlying video features. At the same time, some small variation LRD components may survive after this process. That is why we can see some weak LRD in the trace.

From the above analysis, it can be seen that whether RealVideo is compressed with CBR or VBR, and whether it has LRD or not, the workload has two levels of burstiness: the frames are clustered in small groups, and the packets are clustered within each frame. If a server sends media packets according to the media timestamps, the traffic could be very bursty.

5. MEDIA STREAMING RESULTS

Two issues arise when streaming RealMedia traffic over a wireless LAN: the bursty traffic workload may induce network-layer packet loss, and wireless channel errors may also affect the loss. Both degrade the subjective quality of the streamed media. In this section, we explore these two issuess by streaming RealMedia clips to a wireless client under different channel conditions.

At the subjective level, the playback of the video and audio streams were very smooth for the *Excellent* and *Good* channel conditions. For the *Fair* channel, the playback of the video was jerky, indicating lost video frames, though the visual quality of displayed video frames was good. The sound quality was good too. Under the *Poor* channel condition, the video playback was jerky, some individual pictures were blurry or truncated, and the audio quality deteriorated. In some cases, the attempt to set up the streaming connection failed. All the losses are caused by wireless channel errors (verified with the Sniffer trace). The following subsections provide measurements that help explain the effects observed.

5.1. Network-layer traffic patterns

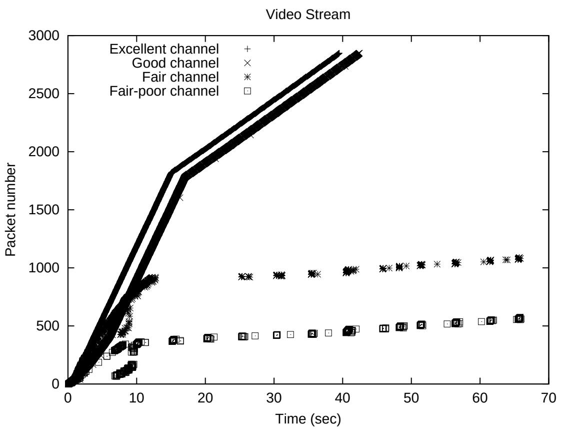
Figure 6 shows the structure of the *Rock Concert* streaming network traffic under different channel conditions. These graphs plot the media packet sequence number versus time, for the video (Figure 6(a)) and audio (Figure 6(b)) streams, respectively. The most obvious observation is that network packets are sent at a consistent rate. The higher rate at the beginning is due to the client's request to fill up the receiving buffer quickly. After the buffer is filled, the streaming rate is reduced to the target rate. The irregularities in the *Fair* and *Poor* channel condition reflect application-layer retransmission.

Other clips have similar streaming patterns. In particular, the *TV Program* was streamed as CBR, although it was compressed as a VBR video. Another observation is that audio packets on the network exhibit a bursty pattern at small time scales. This pattern is evident not only when looking at the *Fair* and *Poor* channels in Figure 6, but also when looking at a zoomed-in version of the plot for *Excellent* and *Good* channels. The graph also confirms our prediction that video packets also have an on-off bursty pattern at small time scales, reflecting the application-layer traffic pattern at the network layer.

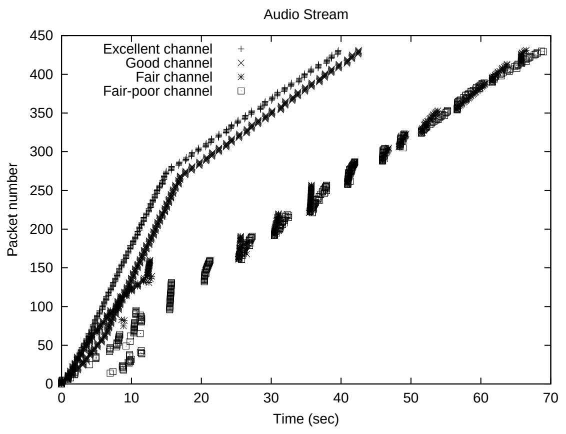
Figure 7 shows the histograms of the inter-packet times of video (Figure 7(a)) and audio (Figure 7(b)) streams under the *Excellent* channel condition. (Note that the y-axis is log scale). We use the *TV Program* clip in this case because its length reduces the effect of the start-up streaming rate on the results. Examining the graph, three clusters of inter-packet times are evident: one near 0 ms, one near 30 ms, and one around 380 ms. We seek to explain each of these peaks.

The peak at 30 ms simply indicates the elapsed time between sending packets into the socket buffer. Given that the total streaming bit rate is 225 kbps and the mean video packet size is 842 bytes, the time needed to send an average-sized packet is about 30 ms (842*8/225,000).

We believe that the peak near 0 ms is related to the many inter-packet times in the 40-100 ms range. In fact, the counts here are roughly equal to those in the 0 ms bin. Since the maximum observed packet size of 1008 bytes takes at most 1008*8/225,000=36 ms to send, the 40-100 ms inter-packet times indicate that RealServer must send multiple packets (two or more) to the socket at a time (probably to reduce system call or scheduler overhead). This accounts for the peak near 0 ms. The 40-100 ms time represents the time RealServer waits for the packets to drain, before doing the next send.

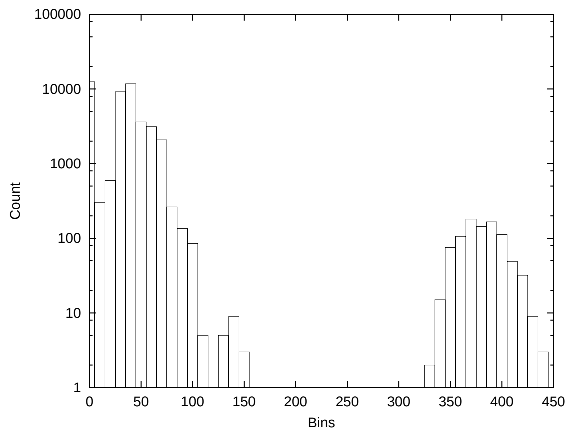


(a) Video

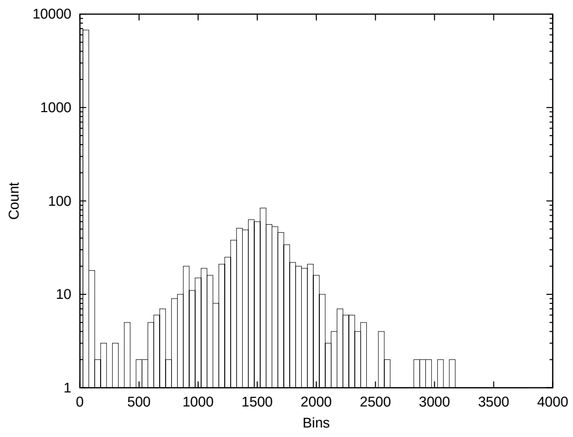


(b) Audio

Figure 6. Structural Characteristics of RealMedia Streaming Network Traffic (*Rock Concert*)

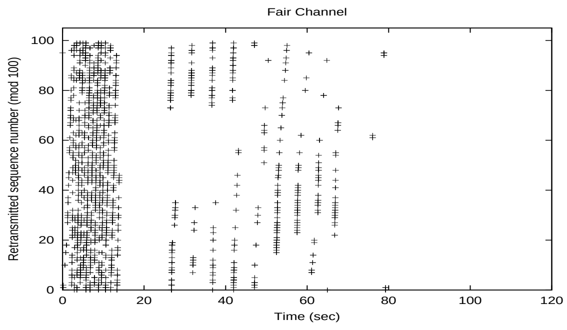


(a) Video Stream

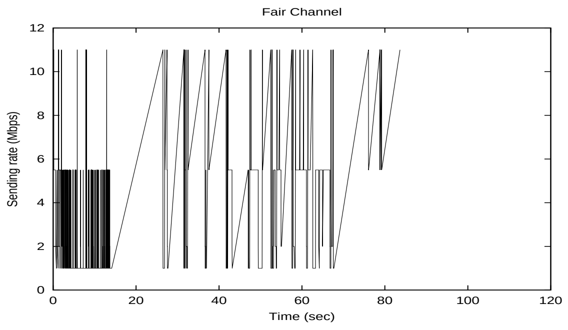


(b) Audio Stream

Figure 7. Histogram of Inter-Packet Time (*TV Program*)



(a) MAC-layer Retransmissions



(b) Access Point Data Rate

Figure 8. Illustration of Impact of WLAN Channel Conditions (Fair Channel)

We suspect that the peak around 380 ms represents the time interval that RealServer stops sending video packets while it is processing audio data. The variations around the peaks are caused by three factors: RealVideo packets are variable-size; RealServer sometimes sends more than one packet to the socket at a time; and the processing time within the operating system kernel varies.

Similarly, Figure 7(b) shows that the audio stream has three cluster peaks at 0 ms, 40 ms, and 1550 ms. Recall that all audio packets are 640 bytes in size, leading to a per-packet time of 640*8/225,000 = 23 ms. The peak near 40 ms occurs because RealServer writes two audio packets into the socket and then waits for twice the regular inter-packet time. In every frame there are 16 audio packets, so the total time is 640*16*8/225,000=364 ms. This value is consistent with the time that the RealServer stops sending video data (see the discussion of the 380 ms peak in the previous paragraph).

From the foregoing discussion, a hierarchical model could be derived for streamed RealAudio and RealVideo traffic: at large time scales (minutes), the overall bit rate is constant; at intermediate time scales (seconds), on-off patterns represent the interleaving of audio and video data; and at fine-grain time scales (sub-second), back-to-back packet trains separated by time gaps represent the "packet batching" behavior of the server to reduce system call overhead. The RealServer has changed the application layer workload dramatically.

5.2. Effect of wireless channel

5.2.1. Error loss behavior and recovery

The error characteristics of a wireless channel affect higher-layer protocols. To study errors, we focus on the MAC-layer retransmission behavior, since each MAC-layer retransmission indicates an error in either the data packet or its ACK.

Figure 8 presents our analysis of MAC-layer retransmissions in the *Fair* channel condition. Figure 8(a) presents a two-dimensional time series representation of MAC-layer retransmissions, with time on the horizontal axis, and MAC frame sequence number (modulo 100) on the vertical axis. Each '+' in the plot represents a retransmitted MAC-layer frame.

The results in Figure 8(a) show bursty error conditions, since retransmissions are clustered both horizontally in the time domain, and vertically in the sequence number domain. In many cases, there are many consecutive MAC frames that require retransmission, particularly in the *Fair* and *Poor* channel conditions. In the *Poor* channel conditions, 67.5% of the media packets sent require at least one retransmission. While the retransmission rate is high, the MAC-layer retransmission strategy is able to recover most of the missing packets without exceeding the MAC-layer retransmission limit.

Figure 8(b) shows the data rate that the AP indicates in the MAC header of each transmitted frame. Comparing the two graphs shows that (as expected) the transmission rate used is inversely related to the channel error rate. When the error rate is high, the transmission rate is low, and vice versa.

Although the 802.11b MAC layer is able to hide most of the wireless channel errors from higher layers, errors still affect the higher layers, in several ways. First, even if a packet reaches the client after MAC-layer retransmission, it takes longer, which affects the application's view of the network delay. Second, the physical layer's transmission rate is automatically adjusted, and this affects the application's view of the channel. Third, some errors are still left to the application layer to solve, which can trigger application-layer retransmission.

The residual errors are further recovered by RealSystem's NACK-based retransmission. Measurement data shows that NACK-based retransmission is reasonably efficient in correcting errors. For example, with the *Fair* channel condition, 133 of the 148 retransmissions requested by the client were successful (89.9%). For the *Poor* channel condition, NACK-based retransmission is slightly less effective: 68 out of 96 retransmitted audio packets (70.8%) were successful, and 132 out of 192 video packets (68.8%).

Ignoring timing issues, the stream quality can be characterized by the "final" effective loss after MAC-layer and application-layer retransmission. The effective loss is zero for *Excellent* and *Good* channels. For the *Fair* channel, one audio packet was lost (0.2%) and 14 video packets (1.3%). This is why the individual picture quality is good. The effective loss for the *Poor* channel is 28.3% (163 packets) for video packets, which explains the blurred video quality. The effective loss for audio packets is 6.9% (30 packets), enough to degrade audio quality.

5.2.2. Application-layer streaming rate

Referring back to Figure 6, in the *Excellent* channel condition, the RealServer starts with a constant 600 kbps video streaming rate. Once the RealPlayer client buffer fills (after 15 seconds or so), the RealServer switches to a 200 kbps video streaming rate (the target rate of this video stream), and maintains that rate for the duration of the session. The streaming structure is similar for both the video and audio streams, though the overall data rate of each is different (note the different scales on the vertical axes in the two graphs).

The behavior for the *Good* channel condition is qualitatively similar, though the errors experienced early in the session temporarily limit the streaming rate requested by the RealPlayer. Once these errors are handled, rates of 600 kbps and 200 kbps are used throughout the session.

For the *Fair* and *Poor* channel conditions, the situation is quite different. In particular, the video streams have a lower streaming rate, well below the target rate. The high error rates cause the RealPlayer to request a lower streaming rate. In many cases, the RealServer has to skip some media packets in order to meet the requested rate budget. This causes jerky video playback. Given that the achievable throughput on the *Fair* channel is about 2 Mbps (see Table 2), this suggests there is room for improvement in the rate adaptation algorithm.

The audio streams, on the other hand, achieved their target rates for all channel conditions, suggesting that RealPlayer gives precedence to the audio stream. In all cases, the AP transmitted all 432 audio packets. The slopes of the plots in Figure 6(b) all match the target rate for the audio stream.

6. SUMMARY AND CONCLUSIONS

In this paper, we conducted a measurement study of RealMedia streaming over an IEEE 802.11b WLAN. Four different RealMedia clips were used in the study, with durations ranging from 1 minute to 2 hours. Traffic data were collected using `tcpdump` and Sniffer Pro 4.6 wireless analyzer. Our study focuses on a hierarchical view of the system: the application-layer view (i.e., the output of the audio/video encoder), the network-layer view (i.e., the departure process for network packets emanating from the RealMedia server), and the wireless channel view. The interactions among these layers affect the whole system performance.

We have studied the interactions between bursty RealMedia application workload and wireless channel errors. The RealAudio codec generates pseudo-CBR traffic, with burstiness at small time scales. In both VBR and CBR, the burstiness at fine-grain time scales is similar to that in RealAudio traffic. In fact, the video traffic has two distinct types of burstiness: groups of frames are generated together, and packets from the same frame are grouped together. Consistent with previous results in the literature, there are strong LRD components found in the VBR-compressed video clips. However, there is little or no LRD in CBR video.

Despite the challenges of the wireless environment, the general wireless streaming quality is quite robust for all but the poorest channel conditions. Several factors contribute to these good results. First, RealServer changes the application workload characteristics dramatically. Instead of sending out bursty traffic, it sends the media as Constant-Bit-Rate (CBR). This "smoothing" effect could reduce the buffer loss in the server and the AP. Second, while the wireless channel has bursty error characteristics, the 802.11b MAC-layer retransmission mechanism is able to hide most physical-layer channel errors from higher-layer protocols. Third, when needed, the application layer's NACK-based error control is effective in recovering the residual missing packets.

Separate experiments[11] have studied the interactions between RealMedia traffic and competing network traffic. In the presence of background traffic, RealPlayer does not compete fairly with TCP connections. When the bandwidth is scarce, the RealPlayer maintains its target rate, to the detriment of TCP flows.

ACKNOWLEDGMENTS

Financial support for this research was provided by iCORE (Informatics Circle of Research Excellence) in the Province of Alberta, and by the Natural Sciences and Engineering Research Council of Canada, through NSERC Research Grant OGP0121969. The authors thank the anonymous reviewers from ITCOM 2002 for their constructive comments on an earlier version of this paper.

REFERENCES

1. M. Chesire, A. Wolman, G. Voelker, and H. Levy, "Measurement and analysis of a streaming-media workload," *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, March 2001.
2. A. Wolman, G. Voelker, N. Sharma, N. Cardwell, M. Brown, T. Landray, D. Pinnel, A. Karlin, and H. Levy, "Organization-based analysis of web-object sharing and caching," *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, October 1999.
3. J. Beran, R. Sherman, M. Taqqu, and W. Willinger, "Long-range dependence in variable-bit-rate video traffic," *IEEE Trans. Commun.* **43**, pp. 1566–1579, February/March/April 1995.
4. M. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar VBR video traffic," *Proceedings of ACM SIGCOMM'94*, September 1994.
5. M. Krunz and S. Tripathi, "On characterization of VBR MPEG streams," *Proceedings of ACM SIGMETRICS*, pp. 192–202, May 1997.
6. A. Mena and H. Heidemann, "An empirical study of RealAudio traffic," *Proceedings of IEEE INFOCOM*, pp. 101–110, March 2000.
7. Y. Wang, M. Claypool, and Z. Zuo, "An empirical study of RealVideo performance across the Internet," *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, pp. 295–309, November 2001.
8. D. Loguinov and H. Radha, "Measurement study of low-bit-rate Internet video streaming," *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, pp. 281–293, November 2001.
9. RealNetworks, "RealSystem production guide: RealSystem release 8 with with RealProducer 8.5," **http://service.real.com/help/library/index.html**.
10. ANSI/IEEE, "Standard 802.11b - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specificiations: Higher-speed physical layer extension in the 2.4 GHz band," September 1999.
11. T. Kuang and C. Williamson, "RealMedia streaming performance on an IEEE 802.11b wireless LAN," *Proceedings of IASTED Wireless and Optical Communications Conference*, pp. 306–311, Banff, AB, Canada, July 2002.
12. T. Kuang and C. Williamson, "A measurement study of RealMedia audio/video streaming traffic," *Proceedings of SPIE ITCOM 2002* **Internet Performance and Control of Network Systems**, pp. 68–79, Boston, MA, July 2002.
13. F. Fitzek and M. Reisslein, "MPEG-4 and H.263 video traces for network performance evaluation," *IEEE Network* **15**, pp. 40–54, November/December 2001.
14. ANSI/IEEE, "Standard 802.11b - part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specificiations," 1999.

15. D. Eckhardt and P. Steenkiste, "Measurement analysis of the error characteristics of an in-building wireless network," *Proceedings of ACM SIGCOMM*, August 1996.
16. R. Agarwal, J. Ayars, B. Hefta-Gaub, and D. Stammen, "Internet draft draft-heftagaub-rmff-00.txt: Real-media file format," March 1998.
17. RealNetworks, "Working with RealProducer 8 codecs," **http://service.real.com/help/library/index.html**, May 2000.
18. RFC2326, "RFC2326: Real time streaming protocol (RTSP)," **http://www.ietf.org/rfc/rfc2326.txt**, April 1998.
19. RFC1889, "RFC1889: A transport protocol for real-time applications," **http://www.ietf.org/rfc/rfc1889.txt**, January 1996.
20. netperf, "Netperf," **http://www.netperf.org**.
21. P. Abry and D. Veitch, "Wavelet analysis of long range dependent traffic," *IEEE Transactions on Information Theory* **4**, pp. 2–15, January 1998.