Performance Analysis of Home Streaming Video Using Orb

Rabin Karki, Thangam Seenivasan, Mark Claypool and Robert Kinicki
Computer Science Department
Worcester Polytechnic Institute
Worcester, MA 01609, USA
{rabin|thangam|claypool|rek}@cs.wpi.edu

ABSTRACT

A new paradigm in video streaming is emerging, that of personal video servers in the home streaming video to remote clients on the Internet. The potential impact of such technologies demands careful study to assess performance and impact. This project studies one such personal video streaming system called *Orb* in a closed network environment, allowing us to control network bandwidths. Our performance evaluation focuses on Orb's method of bandwidth estimation, video performance and bitrates, and resource usage during transcoding. Analysis shows Orb uses simplistic, but effective, methods of determining available bandwidth, dynamic temporal and spatial scaling, and significant CPU cycles when transcoding. The results should be useful for subsequent comparison with other home streaming technologies and capacity planning for network providers.

Categories and Subject Descriptors: C.2.m [Computer-Communication Networks]: Miscellaneous

General Terms: Measurement, Performance.

Keywords: Streaming, Orb, Video.

1. INTRODUCTION

Video streaming over the Internet has grown in popularity, representing the largest fraction of Web-based traffic to the home [2]. While streaming has predominantly been from servers on the Internet down to the home, a new trend is emerging where home users run systems that stream video up from the home to the Internet. While traditional video streaming systems run from well-connected servers, home streaming systems are affected by asymmetric broadband connections typical for many residences. In addition, while typical video streaming systems have well-provisioned servers, home streaming systems may run on commodity hardware with software that may not be configured for streaming. The potential of a multitude of home users streaming video up from their residences demands careful study of these new technologies to assess performance and impact. In particular, careful studies are needed to determine how such technologies ascertain available bandwidth, how they respond to network configurations in terms of capacity use, and how effective they are at producing quality video to clients upstream from the home.

Orb[1] is a free software system which supports streaming video from the home. First launched in 2005, Orb now has more than 7 million registered users.[2] Being more general than a simple streaming server, Orb enables access to media files from a home PC to any supported device (typically a PC with a Web browser) connected to the Internet. Orb allows users to access photos, audio and video, Internet radio, broadcast television (if the home PC has TV tuner card) and even live video streaming from a Webcam. The easy availability (free) and flexibility (only a commodity PC is needed to host) of Orb make it an ideal candidate for the first investigations of home streaming technologies.

This project provides a preliminary investigation into video streaming using Orb. The focus is to: 1) ascertain how Orb determines the bandwidth available for streaming, 2) measure network traffic under different bandwidth constraints, 3) investigate video performance at the streaming client for the same bandwidth constraint and 4) understand resource usage at the streaming host. Experiments were run in a controlled test environment, allowing for fixed network configurations and repeatable results. Specifically, low quality and high quality videos were streamed over network capacities ranging from 250 to 1000 kbps, with measurements at the host to determine resources used during transcoding (converting video from one format to another in real-time) and measurements at the client to determine streaming video performance.

Analysis of the results shows Orb uses a simple, but effective, brief bulk upload and download technique at initialization to determine the available bandwidth. When the available bandwidth is less than the original encoding rate, Orb scales the quality of the video using both temporal and spatial scaling so as to continue smooth playout with re-buffering. This scaling is relatively smooth for low quality videos but varies considerably for high quality videos. Curiously, Orb appears to scale videos up when the bandwidth is available, even if the original encoding does not support these higher rates. Transcoding uses significant server resources, particularly for some video formats.

The remainder of this paper is as follows: Section 2 pro-

NOSSDAV'10, June 2–4, 2010, Amsterdam, The Netherlands.

[1] http://www.orb.com
[2] http://www.orb.com/en/orb_celebrates_bastille_day_with_seven_million_users

vides background on Orb installation, usage, and modes of streaming; Section 3 describes the experimental setup used to evaluate Orb performance; Section 4 presents the analysis of the experimental results; Section 5 summarizes our conclusions; and Section 6 presents possible future work.

2. ORB

In a typical Orb system (depicted in Figure 1), a user installs Orb on a home computer which acts as the *Host* PC. The user modifies the Orb settings to specify the location of the media files stored on the Host. The media files can then be streamed over the Internet to any remote host with an Orb *Client*. The Client can be a stand alone application on a specialized device (such as a app on an IPhone) or simply a plugin in a Web browser. This study considers only using a Web browser on the Client and leaves investigating the performance of a stand alone Orb Client for future work. Both the Client and Host PC register with the Orb *Server*, a dedicated server on the Internet.

At the Client, the user logs into the Orb Server with an account and password that enables access to streaming via the Host. Orb supports a variety of streaming formats, such as Windows Streaming Media, Real Media, Apple QuickTime and Adobe Flash, with configurable preferences set at the Client. To play the video in the chosen streaming format, the Client must have the corresponding codec and player installed. For example, if the Client is on a Windows PC, the user may choose Windows Streaming Media (WSM) as the preferred video format.

Orb supports two modes of streaming, *direct streaming* and *indirect streaming*, depicted in Figure 1.

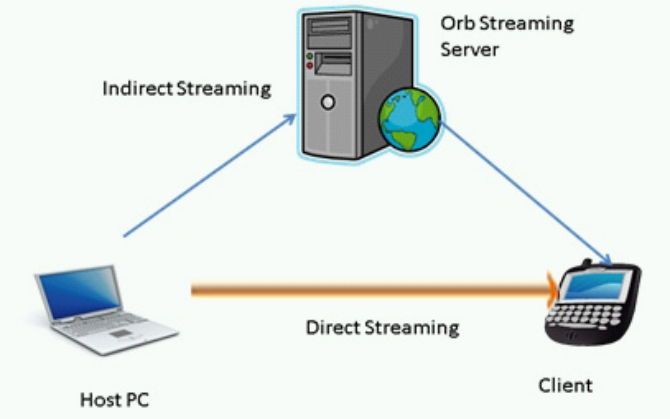


Figure 1: Streaming modes in Orb.

Orb defaults to streaming the video directly from the Host PC running Orb to the Client without going through any intermediate Orb servers. Direct streaming is used in the following scenarios: 1) The Host PC is directly connected to the Internet with a public IP address; 2) The Host PC is connected to the Internet via a router which supports Universal Plug and Play (UPnP) that provides Client access to the Host through a NAT device; or 3) The Host PC and the Client are on the same LAN, even if they share the same public IP address to access the Internet.

When Orb cannot support direct streaming (e.g. the Host PC is hidden behind a NAT device), Orb adopts indirect streaming. During indirect streaming, data is sent from the Host PC to the Orb Server where it is redirected to the Client. Our experiments concentrate on direct streaming, the default Orb behavior, with indirect streaming left as future work.

3. EXPERIMENTS

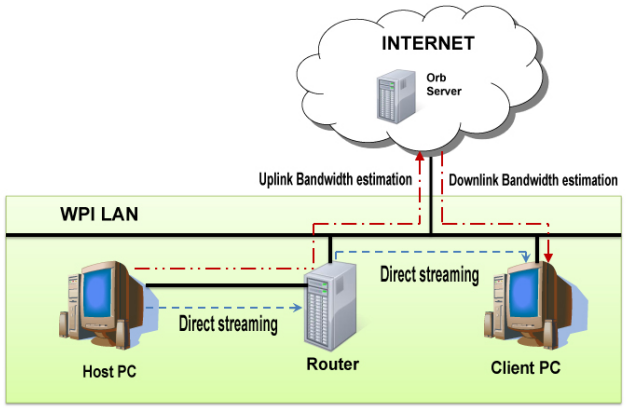


Figure 2: Experimental Setup of Orb Host PC and Client.

Figure 2 depicts the experimental setup used to measure Orb performance. The Host PC and the Client are desktop machines running Windows XP. The Host PC has a Pentium 4, 2.8 GHz CPU with 1 GB RAM, running Orb version 2.51.0032. The Client has a Pentium 4, 2.8 GHz CPU with 1 GB RAM.

The Router is a PC with a Pentium 4, 2.0 GHz CPU and 512 MB RAM running SuSE Linux 10.3 (kernel version 2.6). The Host PC is connected to the Router via a crossover cable, with the Router setup to use Proxy ARP and forward traffic to/from the Host PC. The Router uses the prebuilt *Netem* module that controls the bandwidth to and from the Host PC.

Since Orb requires initial communications between the Host PC and the Client to the Orb Server for login authentication before streaming commences (even in the case of direct streaming), all machines in our experimental setup require access to the Internet. Thus, for our setup, all the three machines, the Host PC, Client and Router, are on the same LAN. The LAN is an Ethernet with a capacity of 100 Mbps and access to the Internet via a 500 Mbps link.

Using this configuration, the performance of Orb streaming videos using TCP is analyzed under controlled network conditions. The unimpeded network has an available bandwidth between the Host PC and the Client of approximately 80 Mbps. The bandwidth between the Host PC and the Client is limited using a token bucket filter in the Netem classless queuing discipline. For example, the commands:

```
tc qdisc add dev eth1 root handle 1:0 netem delay 0ms
tc qdisc add dev eth1 root tbf rate 0.1mbit burst 5kb
   latency 70ms
```

set the maximum bandwidth available to 100 kbps, with a queue size of 5 Kbytes and a maximum latency of 70 ms. The bandwidth was varied across 100 kbps, 250 kbps, 500 kbps and 1000 kbps in subsequent experimental runs. The

ability of Netem to accurately control bandwidth (as well as loss and latency) was verified using Wireshark,[3] Iperf[4] and ping.

A high definition video[5] from the Apple sample videos page was selected and trimmed to 150 seconds, providing a 1280x720 resolution video at 25 fps, encoded with a bitrate of 7 Mbps in H.264 format.

To evaluate streaming and transcoding from multiple, initial source formats, the video was then converted using `ffmpeg`[6] to the following formats: A) Windows Streaming Media low quality: extension .wmv; video - wmv2 codec at 320x240, audio - wma2 codec; 768 kbps; B) Windows Streaming Media high quality: extension .wmv; video - wmv2 codec at 1280x720, audio - wma v8 codec; 1546 kbps; or C) Flash Video: extension .flv; video - flv1 codec at 320x214, audio - mp3 codec; 320 kbps.

Given the prevalence of Microsoft operating systems on end hosts, the videos were encoded in Windows Streaming Media format. Both a low quality version (Section 4.2) and a high quality version (Section 4.3) were used.

To analyze Orb network traffic, Wireshark was installed on both the Host PC and the Client. Wireshark enables analysis of transport protocols used and data rates. To capture video performance, MediaTracker[7] was installed on the Client [1]. For each video played, MediaTracker logs video performance information including: encoding data rate, playout bitrate, buffer progress, video frame rate, video frames lost, video frames skipped, packets lost and packets recovered.

4. RESULTS

This section presents results from the Orb experiments, starting first with studying how Orb determines the available bandwidth from the Host PC to the Client (Section 4.1), proceeding to analysis of streaming with low quality video (Section 4.2) and high quality video (Section 4.3), and ending with transcoding resource usage.

4.1 Bandwidth Estimation

Wireshark traces were used to analyze Orb's bandwidth estimation techniques. Analysis of the traces shows that after a user authenticates with the Orb Server, Orb automatically calculates downstream bandwidth to the Client and upstream bandwidth from the Host. However, the user can explicitly request that Orb recalculate these values and can even manually set them after the automatic calculations.

To calculate the downlink bandwidth, the Client downloads about 4 Mbits of data from the Orb Server. To calculate the uplink bandwidth, the Host PC uploads about 0.4 Mbits of data to the Orb Server. Orb then computes the bandwidth estimate based on the amount of time it takes to transfer the data. The same amount of data, 4 Mbits and 0.4 Mbits were transferred for downlink and uplink bandwidth estimation respectively for all the runs. Upon streaming, Orb uses the minimum of the downstream and upstream values to appropriately scale the quality of the video.

[3]http://www.wireshark.org/
[4]http://sourceforge.net/projects/iperf/
[5]http://movies.apple.com/movies/us/hd_gallery/gl1800/-720p/bbc_cctv_720p20070914.mov
[6]http://www.ffmpeg.org/
[7]http://perform.wpi.edu/real-tracer/#mediatracker

To measure the efficacy of Orb's bandwidth estimation, the actual bandwidth between the Host PC and Client was varied using Netem and the upstream bandwidths reported by Orb were recorded. After running Orb, Iperf was also run in TCP mode to get a baseline comparison of the bandwidth achievable by TCP. Iperf transmits data for 10 secs to estimate bandwidth (i.e. Iperf transmits 4.9 Mbits of data when it estimates bandwidth as 500 kbps). As Figure 3 shows, the bandwidths measured by Iperf and Orb are quite similar, and both are consistent with the bandwidth set by Netem. However, as the available bandwidth goes beyond 2 Mbps, Orb does not report it accurately, displaying a seemingly arbitrary value between 2 and 4 Mbps. It may be that the bottleneck beyond 2 Mbps is upstream on the Internet beyond our Router.

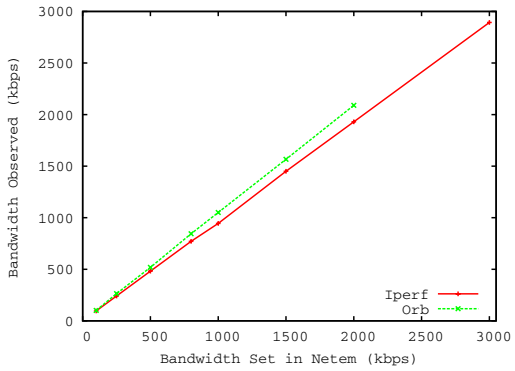


Figure 3: Measured Bandwidth vs. Set Bandwidth.

4.2 Low Quality Video

The first set of experiments used a 320x240 resolution video of total size 14.4 MBytes with a duration of 150 seconds (video A). The encoded bitrate of this video was 768 kbps, calculated as the ratio of the file size to the duration of the video. The video was streamed to MediaTracker on the Client. To vary the streaming session environment, during subsequent runs, the upstream bandwidth of the Host PC was limited to 125 kbps, 250 kbps, 500 kbps and 1000 kbps using Netem.

Figure 4 shows the frame rate for the low quality video with different bandwidth settings. When the available bandwidth is 250 kbps, the frame rate is reduced to 10 frames per second (fps) resulting in a visually degraded video. When the available bandwidth is 500 kbps and above, the video is streamed at 20 fps. This suggests Orb uses temporal scaling to adapt to the available bandwidth. The jump from 10 fps to 20 fps means the scaling levels are fairly coarse. Figure 5 shows the encoded bitrate of the video at different bandwidth settings. Orb adapts the encoded bitrate to the available bandwidth, with the 250 kbps setting yielding an extremely low encoded bitrate. The adaptation of the encoded bitrate to at least four different bitrate levels (as opposed to only two levels, 10 fps and 20 fps, for frame rate) suggests Orb is employing quality scaling in addition to temporal scaling.

Figure 6 shows the bandwidth occupied by the video during streaming for different bandwidth settings. With the exception of the 250 kbps setting, Orb scales the video to

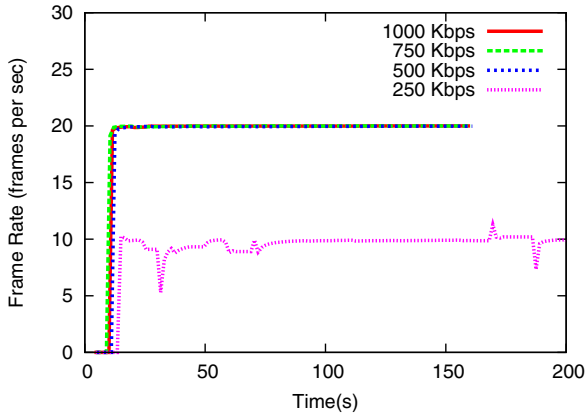


Figure 4: Frame Rate vs. Time (Low Quality Video).

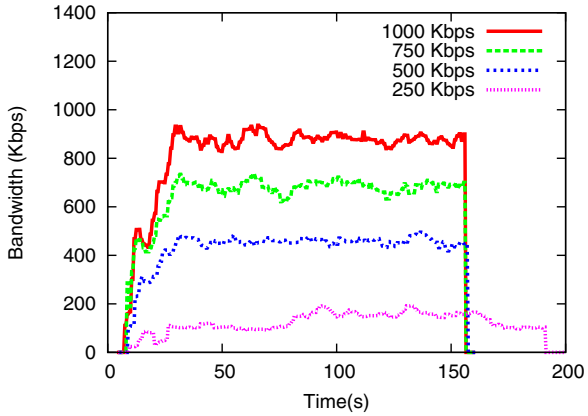


Figure 6: Bandwidth vs. Time (Low Quality Video).

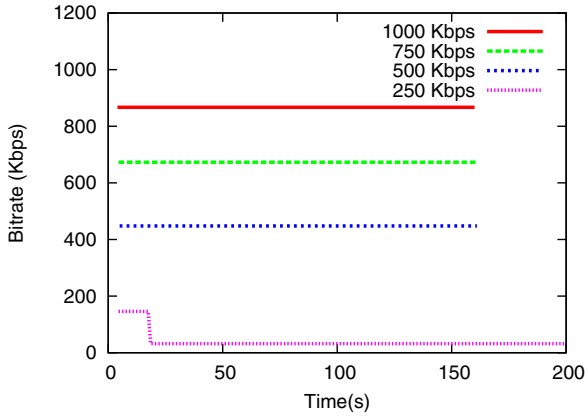


Figure 5: Bitrate vs. Time (Low Quality Video).

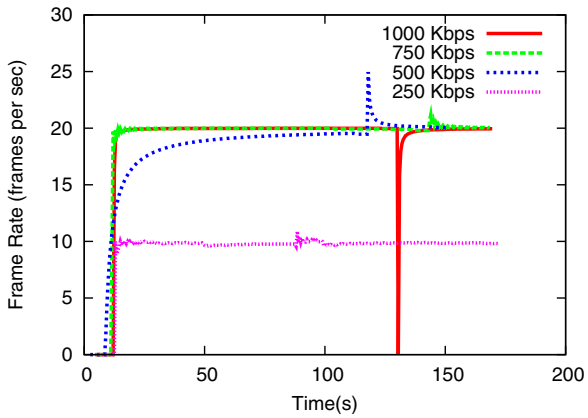


Figure 8: Frame Rate vs. Time (High Quality Video).

stream just below the available bandwidth. When the available bandwidth is more than the encoded bitrate (e.g. 1000 kbps), Orb actually scales *up* the video (beyond the original 768 kbps encoded rate) to use all the available bandwidth. This upscaling is not from the Client trying to download the video as rapidly as possible since the final termination of the video streaming is not shortened, as seen in Figure 5.

Figure 7 plots the buffer progress for the video at different bandwidths. In all cases, the initial playout buffer fills within about 10 seconds. For all cases except at 250 kbps, the playout buffer remains full (100%) for the duration of the session. Only at 250 kbps does the video stop and rebuffer about five times, increasing the video play duration. When the bandwidth is limited to 250 kbps, the playout buffer fluctuates widely. This suggests that the downscaling of the bitrate, seen in Figure 5, is triggered by low buffer values. This adaptation is even more prevalent in the context of the high quality video discussed next.

4.3 High Quality Video

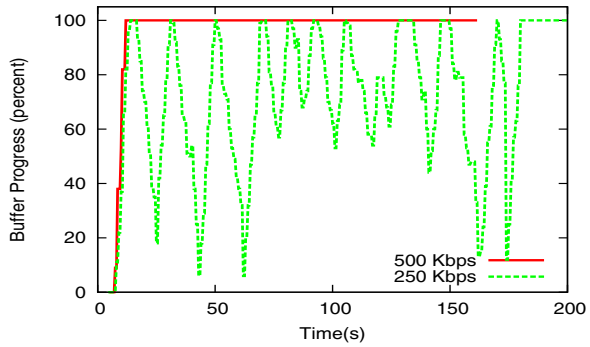
The same experiments were repeated with a high quality video. This video had a 1280x720 resolution, a size 29 MBytes size and a duration of 150 seconds (video B), yielding an encoded bitrate of 1546 kbps.

Figure 8 shows the frame rate for the high quality video over the four bandwidth settings. The frame rates are similar to those for the low quality video with the exception that at 500 kbps, the streaming server takes nearly two minutes to achieve 20 fps and for the 1000 kbps bandwidth setting, there is a noticeable stoppage of playout at the 130 second mark. There are no visually apparent rebuffering events when the frame rate changes.

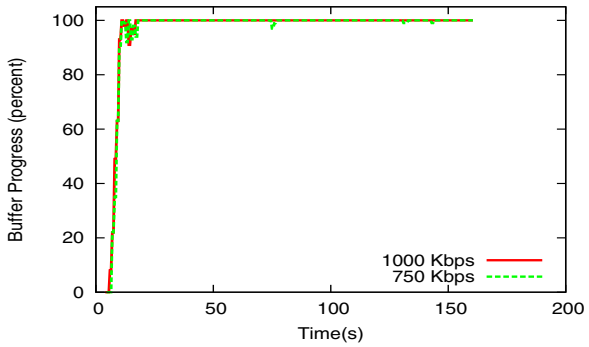
Figure 9 graphs the bitrate for the high quality video under the four bandwidth settings. The encoded bitrates vary considerably and seemingly independently of the frame rates. This suggests quality scaling is used when the video is adjusted mid-stream. In particular, the encoded bitrates appear to be adjusted lower than the available bandwidths in all cases.

Figure 10 shows that the bandwidth used more closely follows the bandwidth settings than do the encoded bitrates. While there exist considerable variations in bandwidth used at all settings, the time durations needed to stream the high quality video are close except for the 500 kbps bandwidth case.

Comparing the buffer progress in Figure 11 to the encoded bitrate in Figure 9 suggests that when the buffer is at 100% for some time, Orb doubles the bitrate, seen for the 500 and 1000 kbps bandwidth settings. At the 1000 kbps bandwidth

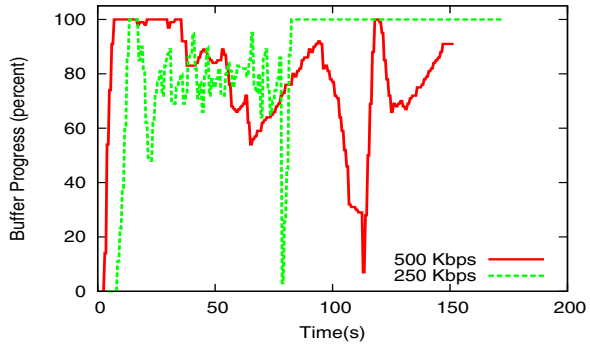


(a) 250 kbps and 500 kbps

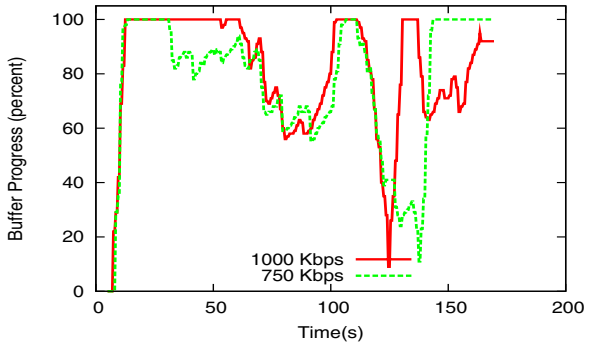


(b) 750 kbps and 1000 kbps

Figure 7: Buffer Progress vs. Time (Low Quality Video).



(a) 250 kbps and 500 kbps



(b) 750 kbps and 1000 kbps

Figure 11: Buffer Progress vs. Time (High Quality Video).

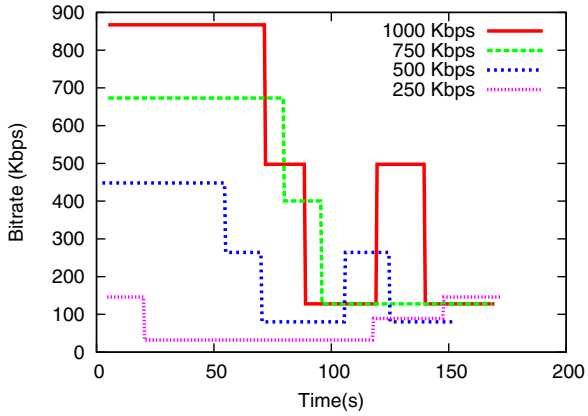


Figure 9: Bitrate vs. Time (High Quality Video).

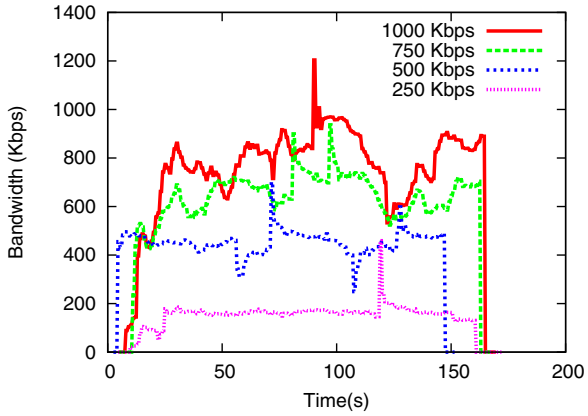


Figure 10: Bandwidth vs. Time (High Quality Video).

setting, Orb starts streaming at a bitrate of around 900 kbps and after about 70 seconds, the buffer progress goes below 60% and Orb reduces the bitrate to 500 kbps. Since the buffer progress does not improve, Orb further reduces the bitrate. At around 110 seconds when the buffer progress has been 100% for some time, Orb doubles the bitrate. However, Orb never doubles the bitrate beyond the initial bitrate it calculated for the bandwidth estimation. The video stopped and rebuffered once for all bandwidth settings with high quality video.

4.4 Host Load

Orb transcodes videos at the Host PC to the format required by the Client in real-time. Orb uses the ffmpeg library (included during Orb installation) executed as a separate process to handle the transcoding. For these experiments, the transcoding process was monitored using the Windows Performance Monitor (perfmon) utility, with data reported for an upstream bandwidth of 1200 kbps. The pro-

cessor on the Host PC has Intel's hyper-threading technology, meaning perfmon reports two separate processors (so the maximum CPU usage is 200%).

	To Flash		To Windows	
Source Video	Bit Rate (kbps)	CPU (%, stddev)	Bit Rate (kbps)	CPU (%, stddev)
A	1125	30, 6	1036	56, 11
B	1130	134, 26	887	165, 17
C	985	34, 13	894	59, 27

Table 1: Conversion to Flash Video or Windows Streaming Media Video.

As seen from Table 1, the bitrate used for all cases is similar, regardless of original source or destination formats. When the video quality is low, the CPU usage is modest, but when the video quality is high, the CPU usage is considerably higher. When converting to Windows Streaming Media as the destination format, CPU usage is considerably higher than when converting to Flash Video format.

5. CONCLUSION

The growth in streaming technologies and broadband access has enabled a new paradigm in streaming to emerge – streaming video upstream from commodity servers and devices located in the home to remote locations on the Internet. Studies are needed in order to better plan for the impact such technologies may have on networks as well as to better understand the strengths and limitations of these home streaming technologies.

This project provides a preliminary performance evaluation of Orb, a home streaming system designed to serve multimedia content directly from a users Host PC to remote Clients on the Internet. Carefully designed experiments control the amount of bandwidth available from our Orb Host and Client, allowing study of the methods for available bandwidth, media scaling and resource usage.

Orb automatically calculates downlink/uplink bandwidth before commencing streaming. Downlink bandwidth is calculated by timing a download of approximately 4 Mbits of data from an Orb Server to the Client, while uplink bandwidth is calculated by timing an upload of approximately 0.4 Mbits of data from the Host PC to an Orb Server.

Orb does both temporal and quality scaling, reducing the frame rate and frame quality in order to provide a bitrate under the available bandwidth. However, the temporal scaling levels and encoded bitrates only coarsely fit the available bandwidth. Interestingly, when the available bandwidth is greater than the encoded bitrate, Orb scales up the bitrate in an attempt occupy the entire available bandwidth, even though the original video encoding cannot make use of the extra bits. Orb starts streaming at the encoded bitrate based on the initial bandwidth estimation. Orb then dynamically adapts the bitrate apparently based on the buffer progress when the video is playing. When the buffer progress drops below a threshold, Orb reduces the bitrate by half. When the buffer progress is 100% for some time, it doubles the bitrate. This coarse scaling driven by fixed buffering suggests there is substantial room for improvement in terms of making better use of potentially limited bandwidth in a residential broadband connection.

Orb does transcoding of the original videos from one format to another, making it resource intensive at the Host PC in terms of CPU usage. The transcoding load is largely dependent on the source video, destination video quality and formats, but conversion to Adobe Flash video format is computationally cheaper than conversion to Windows Streaming Media format. The video transcoding is multithreaded, launching between 15-22 simultaneous threads suggesting merit for multi-core or hyper-threading enabled processors. The resource-intensive nature of transcoding and the possibility that many home users will use resource limited Hosts suggests room for improvement in terms of more sophisticated transcoding techniques as well as informed content selection choices on the Client.

The results of this study should be useful for comparison with other home streaming technologies. It also should provide guidelines for users to provide adequate upstream bandwidth for Orb use. Since Orb allows users to choose different source and destination formats, knowledge of bandwidth and load for the different formats should help users to make informed choices based on their available resources.

6. FUTURE WORK

While this paper provides a first look at one home streaming technology, given the relative newness of home streaming and the lack of exploration in this area, there are numerous areas for future work.

As indicated in Section 2, when an Orb Host is unable to perform direct streaming it falls back to indirect streaming. Indirect streaming uses an additional node to the streaming process, that of the Orb Server, possibly adding new complications to bandwidth usage and video performance. Orb also supports different destination video formats and other media types such as streaming audio that warrant evaluation. While our focus was on Orb as used through a Web browser, dedicated Orb Clients, particularly on mobile platforms, merit study as untethered devices proliferate.

Our study does not investigate the effect of background network load on video performance. Further studies can see how Orb reacts to competing network traffic and whether or not Orb can successfully adapt properly to varying background loads.

Other home streaming technologies to be studied include Slingbox[8] and LocationFree[9]. Unlike Orb in which Hosts run on top of commodity PCs, these alternate technologies stream video from form-factor devices, where the device includes the processor, software and network interface all in one. Sony makes wall-mounted and portable televisions designed to stream video from their LocationFree device.

7. REFERENCES

[1] M. Li, M. Claypool, and R. Kinicki. MediaPlayer versus RealPlayer – A Comparison of Network Turbulence. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, Marseille, France, Nov. 2002.

[2] G. Maier, A. Feldmann, V. Paxson, and M. Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *Proceedings of the ACM Internet Measurement Conference (IMC)*, Chicago, IL, USA, Nov. 2009.

[8] http://www.slingmedia.com/go/slingbox

[9] www.sony.com/locationfree