Thin to Win? Network Performance Analysis of the OnLive Thin Client Game System

Mark Claypool, David Finkel, Alexander Grant and Michael Solano
Computer Science and Interactive Media & Game Development
Worcester Polytechnic Institute, Worcester, MA 01609, USA
email: {claypool,dfinkel}@cs.wpi.edu

***Abstract*—The growth in network bitrates and server-based processing has provided a renewed opportunity for thin client games, where the server does heavy-weight computations, sending only the visual game frames to the client, and the client displays frames, sending only the user actions to the server. Understanding the traffic characteristics of thin client games is important for building traffic models and classifiers and planning network infrastructures to meet future demand. This paper provides the first detailed study of the network characteristics of OnLive, a commercially available thin client game system. Carefully designed experiments measure OnLive game traffic for several game genres, analyzing the bitrates, packet sizes and inter-packet times for both upstream and downstream game traffic, with comparisons to traditional game clients and streaming video. Results indicate OnLive rapidly sends large packets downstream, similar but still significantly different than live video. OnLive less frequently sends much smaller packets upstream, significantly different than traditional game client traffic. The results should be a useful beginning for building effective traffic models and classifiers, and for preparing end-host networks to support this upcoming generation of computer games.**

I. Introduction

The computer game industry has seen tremendous growth in recent years and is forecasted to be over a $100 billion industry by 2015 [1], on par with the U.S. movie industry revenue worldwide (DVD and film).[1] Online games have also seen considerable growth, catalyzed on by the growth in residential broadband Internet connections with high capacities and low latencies that encourage game developers to incorporate networked features into their products.

Thin clients, where the local computer is primarily an input and output device and the remote computer does the majority of the processing, have seen a resurgence in use because today's networks capacities and latencies can support bandwidth-intensive, client-server interactions. Thin clients are expected to grow from over 12 million units shipped in 2011 to over 25 million in 2013, and by 2014, 15% of traditional professional desktop PCs will be replaced by virtual desktops accessed from thin clients [1]. Using thin clients for games has already gone commercial, with companies such as OnLive[2] and GaiKai[3] having expanding commercial success.

In order to classify network traffic and plan infrastructures, it is important for engineers to have knowledge of the network load caused by emerging end-host applications. For traditional network games, traffic from a number of popular games has been characterized to provide suitable traffic models for testing existing or planned network designs. There have been numerous studies of traffic models for popular PC games [2], [3] and even game consoles [4]. There have also been studies of traditional video, both pre-recorded [5], [6] and live [7], [8]. However, to the best of our knowledge, measurements of thin clients across games with comparisons to traditional games and streaming video have not been done.

Using sniffing to capture OnLive traffic, this paper investigates the network characteristics (the size and frequency of data sent and the overall bitrate), which we call *turbulence*.[4] This study seeks to answer the following questions (with a brief answer as revealed by this study provided in parentheses):

1) *What is the network turbulence for OnLive games?* Characterizing the data rates, packet sizes and inter-packet times for OnLive games is a critical first step for building accurate game traffic models and for traffic classifiers. (Answer: OnLive games have high downstream bitrates, about 5 Mb/s with 1000 byte packets, while much more moderate upstream bitrates, about 100 Kb/s with 150 byte packets.)

2) *Does the network turbulence for different game genres (such as first-person vs. omnipresent) differ from each other?* If the answer is no, then research efforts can study traffic on one game genre only, saving costs. (Answer: The characteristics of game traffic are similar for all genres, but total bitrates for downstream and upstream traffic can vary by as much as 50%.)

3) *Does the network turbulence for OnLive games differ from traditional games?* If not, then mature, previously developed models for online games can be used to analyze the impact of thin client games. (Answer: Downstream OnLive traffic is more similar to downstream live video, while upstream OnLive traffic is only somewhat similar to upstream traditional game traffic.)

4) *Does the network turbulence for OnLive change with different network conditions?* If not, then this suggests OnLive is not adaptive to network conditions, making it easier to model but with more potential to disrupt networks. (Answer: OnLive does not appear to adapt bitrates to loss or latency, but does adapt to capacity limits. Frame rates adapt to both capacity limits and loss, but not to latency.)

[1] "How Much Does Hollywood Earn?", Information is Beautiful, http://www.informationisbeautiful.net/2012/how-much-does-hollywood-earn/

[2] http://www.onlive.com/

[3] http://www.gaikai.com/

[4] The term "footprint" is used in the context of the memory size of software. In networks, the size and distribution of packets over time is important, hence our word "turbulence".

The rest of this paper is organized as follows: Section II provides related work on measuring performance of thin clients; Section III describes our measurement setup and methodology; Section IV analyzes our results in relation to the questions posed above; Section V summarizes our conclusions and presents possible future work.

II. Related Work

Lai and Nieh [9] use a novel benchmarking technique to evaluate the performance of several thin client platforms, and Packard and Gettys [10] passively monitor network traffic between X clients and servers under capacity controlled conditions. Among other findings, latency is seen to dominate capacity in limiting performance. The above studies, as well as others (e.g., [11]) provide detailed insights into thin client performance, but predominantly pertain to traditional applications and not necessarily thin clients and games.

Winter et al. [12] propose a thin client system designed specifically for streaming and interactive games. Their system streams screen images after rendering by the graphics card, thus reducing bandwidth and increasing visual quality for streaming video games.

Chang et al. [13] propose a methodology for studying thin-client game systems based on the game as displayed on the server compared to the game as displayed on the client, quantifying frame rate and frame distortion. Similar to work on traditional client-server games [14], the authors find that frame rate is more critical to gameplay than frame distortion.

Our work complements previous work by providing more detailed insights into the characteristics of thin client game traffic, as well as evaluating several representative games.

III. Methodology

A. Application Selection

In order to ascertain if turbulence for OnLive varies with the type of game, representative games from three different genres were selected. Following the game classification described in [15], games were chosen from each dominant genre: *first person avatar*, *third person avatar*, and *omnipresent*. The selection of games was limited to those available via OnLive (about 300 titles at the time of the experiments). Games were chosen based on perceived popularity and with a similar release dates in an attempt to provide for relatively comparable visual graphics quality. Table I summarizes the games selected, with indicated classification and year of initial release. As a measure of the system impact, the minimum system requirements as specified by the game manufacturers is similar for each game, with Batman having a higher RAM requirement than the other two games and only Rome requiring a dual core Intel processor.

TABLE I: Games used in experiments

Game	Nickname	Classification	Release
Unreal Tournament III	UT	First person avatar	2007
Batman: Arkham Asylum	Batman	Third person avatar	2009
Grand Ages: Rome	Rome	Omnipresent	2009

In order to compare OnLive downstream traffic to video, YouTube was selected as a representative candidate of pre-recorded streaming video, and Skype was selected as a representative candidate for live streaming video.

B. Measurement Testbed

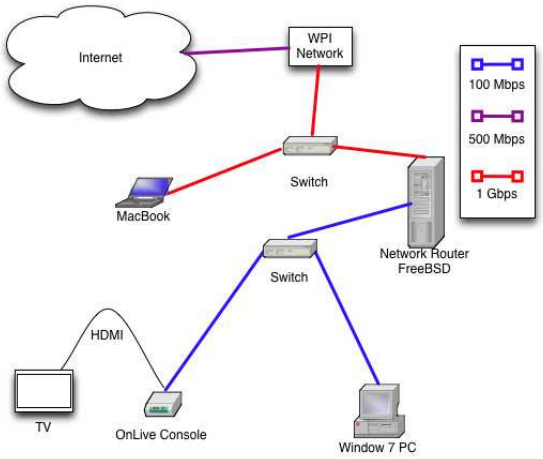

Fig. 1: Measurement testbed

Figure 1 depicts the measurement testbed setup for our experiments. The OnLive MicroConsole runs the games and is connected to an HDTV over HDMI running at 1080p. OnLive specifies a minimum downstream bandwidth of 5 Mb/s to run at this resolution. A desktop PC runs Skype and YouTube for measuring streaming application performance, and Fraps[5] to record application frame rates. The Skype application connects to a MacBook for two-way conferencing. A PC configured as a network router runs FreeBSD and Dummynet [16] to allow emulation of a wide variety of network conditions at the IP level, including network capacity, loss, and latency. The PC runs tcpdump[6] to capture traffic for analysis of turbulence after the experiments complete. The WPI campus egress to the Internet is 500 Mb/s, while switches on the campus have 100 Mb/s and 1 Gb/s capacities.

C. Experiments

Pilot studies captured game traces for varying lengths in order to determine how long individual sessions should run. Based on these runs, it was determined 2.5 minutes of gameplay was long enough for the player to have repeatedly gone through all core game interactions, and for the network behavior to be consistent for all applications.

1) Games: For each game, a scenario was selected to represent core game play.

UT: A free-for-all match on the Rising Sun map was started, containing 10 Bots, no mutators and forced respawns, with no limit on the score or time. The player then competed for weapons, armor, and health using them to defeat the Bots.

Batman: A challenge mode was used, where the player would start in a small square room with 3 enemies, with more appearing

[5]http://www.fraps.com/
[6]http://www.tcpdump.org/

continuously. The player used attack combos to incapacitate as many enemies as possible until the number of enemies was overwhelming.

Rome: A empty map was used where the player constructed buildings using the technology tree, with no enemies encountered. The player built (in order): 3 insulas, 1 pig farm, 1 wheat farm, 1 aqueduct, 1 large water fountain, 1 logging shed, 2 more insulas, 1 butcher shop, 1 farmers market, 1 grape farm, and 2 logging sheds.

The packet captures proceeded from the start until the end of each game session, trimming each session to 2.5 minutes of the core gameplay.

2) Skype: For the Skype testing, all non-essential applications were closed before the Skype video call was made. One participant was on the MacBook and the other on the PC. A short pause was given to establish the call, then the session was set to full screen mode before starting FRAPS and the packet capture.

3) YouTube: For the YouTube testing, a video[7] of an omnipresent game, a two-player session of StarCraft 2, was chosen. The video could not start automatically at 1080p in fullscreen. So, using Google Chrome, the URL link was entered at the same time the packet capture was started, and changed to full screen and 1080p right when it came up.

D. Parameters

All together, three games and three applications were tested over conditions with varied capacity, loss and latency:

Game genre	UT, Batman, and Rome
Application	Game, live video, pre-recorded video
Capacity (down:up)	5:1 Mb/s, 10:2 Mb/s, and unrestricted
Latency (round-trip)	0, 40, and 70 milliseconds
Loss (downstream)	0%, 1%, and 1.5% loss
Iterations	3 runs for each experiment condition

IV. Analysis

A. Turbulence

Our traces show OnLive uses UDP for both downstream and upstream game traffic. Analysis includes UDP and IP headers, unless otherwise noted.

To assess network turbulence, bitrate, packet size and inter-packet times are analyzed.

Figure 2 depicts a comparison of the downstream bitrate (computed every second) for OnLive for the three games under test. In terms of bitrates, all three trendlines look similar. Thus, in this graph, and all subsequent bitrate graphs, only the second trial is shown. The x-axis is the measurement time in seconds, and the y-axis is the bitrate in Kb/s. Each game is depicted by a separate trendline. The similarity in bitrates for UT and Batman (about 6.3 Mb/s) could be because of the relatively similar camera angles they provide to the player, while Rome has around half the bandwidth (about 3.8 Mb/s), possibly because of the different camera angles afforded by an omnipresent game. There is more variance in the bitrate trendlines for Rome than for UT or Batman.

Figure 3 depicts a comparison of the cumulative distribution functions (CDFs) of the downstream packet sizes for the

[7]http://www.youtube.com/watch?v=0NTeyF6wQUs

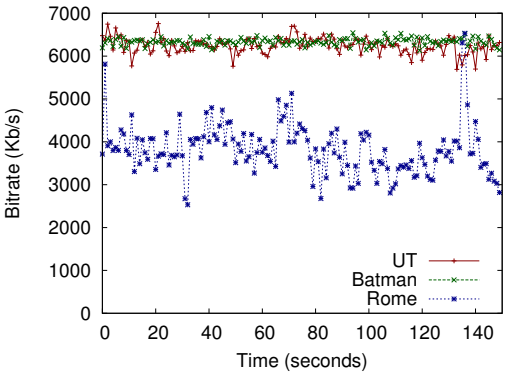

Fig. 2: Downstream bitrate

three games. The x-axis is packet size and the y-axis is the cumulative distribution. Note, for easier readability, only 1 out of every 1000 points are plotted. The trendlines all look similar, probably due to the nature of the encoding of the display images sent from the server to the client. The mode, about 40% of all packets, is 1414 bytes, which is smaller than the MTU of 1500 bytes that could be used. About 10% of the packets are 622 bytes. The other 60% of the packets are distributed fairly uniformly between about 100 and 1400 bytes.

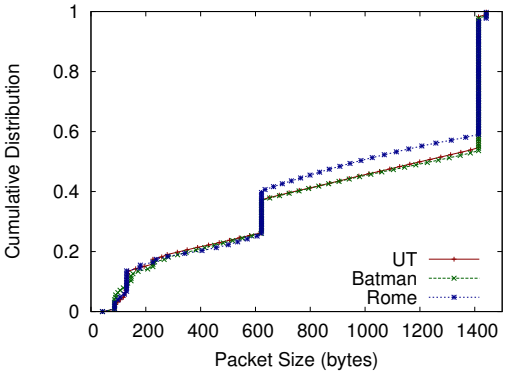

Fig. 3: Downstream packet size

Figure 4 depicts a comparison of the inter-packet time CDFs for the three games, also plotting 1 out of every 1000 points. The x-axis is the time between packets, in milliseconds. In general, inter-packet times are quite small, with a median less than half a millisecond. The maximum inter-packet times are only tens of milliseconds. The trendlines all look similar, particularly so for UT and Batman.

Figure 5 depicts a comparison of the upstream bitrate for OnLive for the three games, showing the second trial (as for the downstream traffic, the other trials were similar). The x-axis is the measurement time in seconds (zoomed in to seconds 50-100 to more easily differentiate the trendlines), and the y-axis is the bitrate in Kb/s with trendlines as in earlier graphs. All three games have considerable variation in their upstream bitrates, ranging from about 50 to 150 Kb/s. UT has a slightly higher upstream bitrate than Batman or Rome, possibly due to the fast-action nature of first person shooter games. Note that the y-axis scale in Figure 2 is 40 times greater than the y-axis

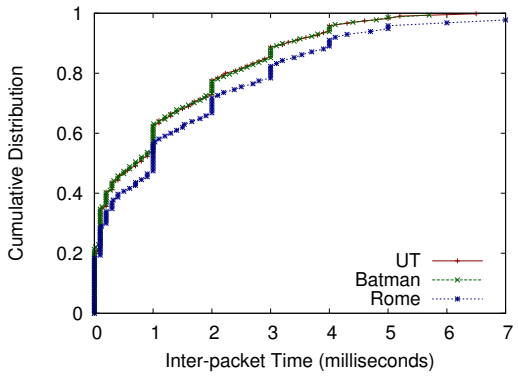

Fig. 4: Downstream inter-packet times

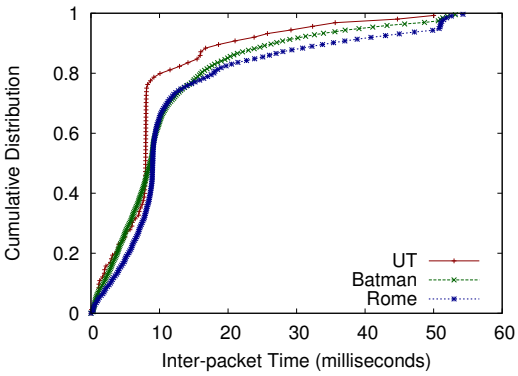

Fig. 7: Upstream inter-packet times

scale in Figure 5, since the downstream to upstream bitrate ratios are about 50 to 1.

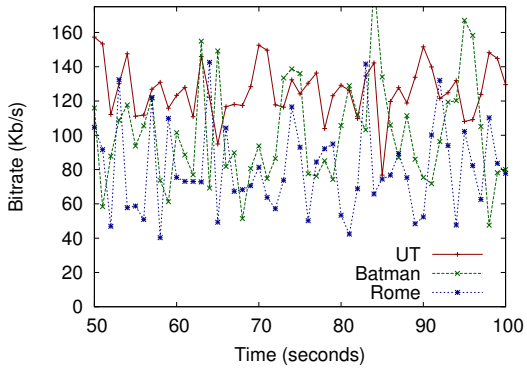

Fig. 5: Upstream bitrate

Figure 6 depicts a CDF of the upstream packet sizes for the three games. Most upstream packets are small, about 100 bytes of application payload after subtracting out the IP headers. Nearly all the packets are under 250 bytes. Note, the x-axis scale is only up to 350 bytes out of a typical 1500 byte MTU.

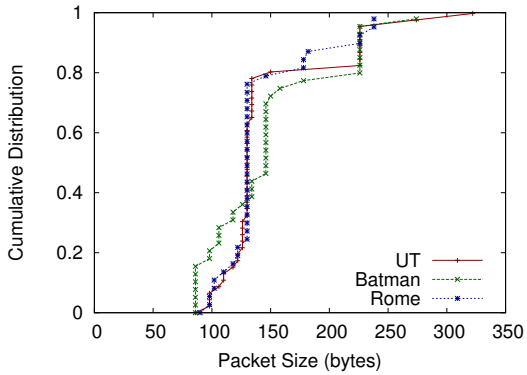

Fig. 6: Upstream packet size

Figure 7 depicts a comparison of the inter-packet time CDFs for the three games, plotting 1 out of every 200 points. The inter-packet times are still small, most under 20 milliseconds, but are considerably larger than the downstream inter-packet times (the x-axis scale for Figure 7 is about 10x larger than for Figure 4). The maximum inter-packet times upstream are slightly over 50 milliseconds. The trendlines all look similar, despite the fact that the player interactions with each game genre may be considerably different.

Table II provides summary statistics (mean values) for the OnLive games, showing both upstream and downstream turbulence. Overall, OnLive traffic is quite asymmetric, with downstream having about 50x higher bitrates, 10x more packets per second and 6x larger packets.

TABLE II: Turbulence for OnLive (up and down, mean values)

Game	Bitrate (Kb/s)	Packet Size (bytes)	Inter-packet Time (msec)
UT	125 6247	146 947	9.5 1.2
Batman	100 6333	152 953	11.7 1.2
Rome	86 3817	143 914	13.4 1.6

B. Network Perturbations

Given the relative similarity in turbulence for the three games studied, independent of genre, for subsequent experiments UT is used as the representative OnLive game. Also, due to space constraints, graphs of packet size and inter-packet times are not shown, only bitrate. Also, as a final focus, since downstream OnLive traffic dominates that of upstream, primarily downstream traffic is analyzed.

Dummynet was used to restrict capacity on the link to the game console to 5:1 Mb/s and 10:2 Mb/s. Figure 8 depicts UT bitrates for 10 Mb/s and 5 Mb/s, with the unrestricted trendline (top) from Figure 2 for comparison. Restricting the bitrate to 10 Mb/s (roughly, that of a typical residential broadband link in the U.S.) on the downlink yields a bitrate of about 4200 Kb/s, with a further restriction to 5 Mb/s dropping the bitrate to just over 2000 Kb/s. There is a slight decrease in variance with increased capacity restrictions. In both restricted cases, the downstream bitrate is about half the capacity restriction.

Dummynet was used to induce 1% and 1.5% packet loss on the link to the game console, and then 40 and 70 msec of added round-trip latency. Figure 9 depicts the corresponding UT bitrate comparisons. There is little apparent effect on bitrate to added latency and loss, except perhaps slightly more bitrate variance with increased loss.

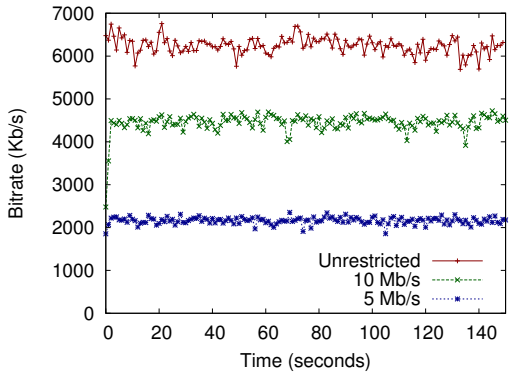

Fig. 8: Downstream bitrate with capacity restrictions

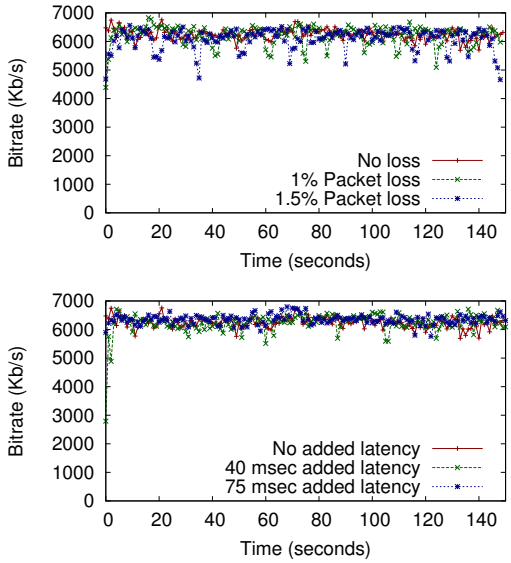

Fig. 9: Downstream bitrate with induced packet loss (top) or latency (bottom)

C. Frame Rate

Since the focus of this paper is on network turbulence, only a summary table of frame rates is shown. Table III depicts the average frame rates recorded for UT under the tested network conditions. Unrestricted, OnLive runs games at 60 f/s, substantially above full-motion video rates but typical of many traditional game clients. Scenarios of induced latencies do not result in reduced frame rates. However, OnLive responds to loss scenarios, both 1% and 1.5%, by playing the video out at reduced frame rates despite the lack of observable change in downstream network bitrates, seen previously in Figure 9. A similar reduction in frame rate accompanies the 10 Mb/s downstream bitrate restriction. With a 5 Mb/s downstream bitrate restriction, OnLive reduces the frame rate even more, down to 25 f/s. Previous work [17] has shown a marked dropoff in player performance for reduced frame rates.

D. Streaming Comparison

This section compares OnLive turbulence to that of Skype and YouTube, and to previously published analysis of traditional games and multiplayer virtual environments. While studies were conducted over the range of network conditions, only the downstream bitrates are analyzed here due to space constraints.

Figure 10 depicts a comparison of the bitrates for YouTube, UT and Skype. YouTube videos are pre-recorded, so unlike in OnLive or Skype, YouTube can potentially download as fast as the network allows. In fact, YouTube finishes the video download in about 90 seconds, so the x-axis is scaled shorter than for previous bitrate graphs. YouTube uses TCP, allowing the bitrate to expand to fill available capacity, while also having more variation due to congestion and flow control mechanisms built into the protocol. Skype has a significantly lower bitrate than OnLive, only about 2200 Kb/s compared with about 6200 Kb/s for OnLive. The bitrate ratio of OnLive to Skype, about 2.8 to 1, is on par with the ratio of their screen resolutions (1080p vs. 720p), about 2.25 to 1. The variances in bitrate for Skype and OnLive look similar.

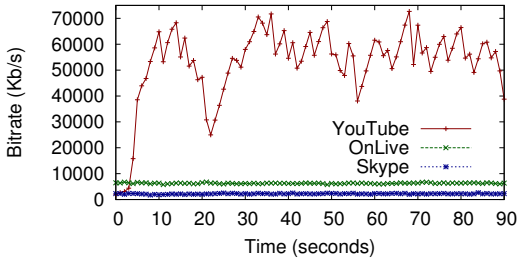

Fig. 10: Downstream bitrate for streaming applications

E. Summary

A final summary seeks to put OnLive turbulence in the context of other interactive applications. A complete comparison for all cases, upstream and downstream, using bitrate, packet size and inter-packet times is difficult given space constraints and, moreover, such analysis would likely not be informative because of the sheer number of details. Instead, median values are summarized for bitrate, packet size and inter-packet times primarily for downstream traffic, often the bottleneck. UT on a PC is chosen as the representative game for comparison, and Second Life is selected as a representation of a multiplayer virtual environment, with complete published analysis from Beigbeder et al. [18] and Kinicki and Claypool [19], respectively.

Table IV presents a comparison of the relevant data.

TABLE III: Frame rates for OnLive games (approximate averages)

Scenario	Frame rate
Unrestricted	60 f/s
40ms latency	60 f/s
70ms latency	60 f/s
1% loss	30 f/s
1.5% loss	30 f/s
10 Mb/s	30 f/s
5 Mb/s	25 f/s

Comparing thin client (OnLive) to traditional network clients, downstream, thin client games have much greater turbulence across all genres, with about 100x greater bitrates, 15x larger packets, and 60x times more frequent packets. This is perhaps expected given that traditional game clients receive game object updates while thin game clients receive game frames. However, even upstream (not shown), thin client games have significantly greater turbulence, with 2x greater bitrates, 2x larger packets, and 5x times more frequent packets. Thus, any switch to thin client games from traditional client games must take into account a significant change in the network traffic.

In general, traditional network games have relatively low turbulence, considering bitrates, packet sizes and inter-packet times. Virtual environments have an order of magnitude more turbulence, reflecting the flexible, dynamic nature of the interactions. Live video has still three times more turbulence, owning to the frequent updates required to display video in real-time. While pre-recorded video has variable turbulence since there are fewer time constraints on clients watching the video, pre-recorded video expands to meet available capacity. Thin client games appear most similar in turbulence to live video, requiring frequent transmission of large packets in order to maintain a smooth frame rate depicting the game. This large turbulence, coupled with the real-time nature of computer games, suggests meeting the quality of service requirements of thin client games is a challenge in terms of network planning similar to that of high-quality video conferencing.

TABLE IV: Downstream turbulence for online applications (medians)

Application	Bitrate (Kb/s)	Pkt Size (bytes)	Inter-Pkt (msec)
Traditional Game	67	75	45
Virtual Environment	775	1027	9
Live Video	2222	1314	0.1
Thin Client Game	**6247**	**1203**	**0.7**
Pre-recorded Video	43914	1514	0.1

V. Conclusions

The growth in connectivity and capacity of networks presents the opportunity for thin clients, clients that primarily handle input and output and not computation, to be used for computer games. Understanding the network traffic characteristics, the *turbulence*, is an important component for designing systems to support this new kind of traffic.

This paper provides a detailed study of the network turbulence of a prominent, commercial thin client game system – *OnLive*. Carefully designed experiments allow for comparison of network turbulence across game genres and streaming video applications and a variety of network conditions. Leveraging previous research allows comparison with traditional network game turbulence.

Analysis shows OnLive traffic has downstream turbulence similar to high-definition, live video, with large, frequent packets and high bitrates. OnLive upstream traffic has far less turbulence, but still significantly higher bitrates and packet rates than traditional upstream game traffic. OnLive turbulence does not respond to network perturbations much, but does adapt to changes in downstream capacity and in the frame rate provided to players when there is packet loss.

Future work can include testing additional games, possibly expanding the selection in the genres chosen or selecting other genres. Other thin client game systems can be evaluated, such as GaiKai, recently purchased by Sony. Research to classify and then potentially treat thin client game traffic can build directly upon the results and analysis presented here, with impacts assessed for actual thin client game traffic. The measurement data can be used to build thin client game traffic models. The traces from our experiments are available at:

`http://perform.wpi.edu/downloads/#onlive`

References

[1] F. Biscotti, B. Blau, J.-D. Lovelock, T. H. Nguyen, J. Erensen, S. Verma, and V. Liu, "Market trends: Gaming ecosystem," Gartner, Inc., 2011, ID Number: G00212724.

[2] W.-C. Feng, F. Chang, W.-C. Feng, and J. Walpole, "Provisioning Online Games: A Traffic Analysis of a Busy Counter-Strike Server," in *ACM Intrnt Meas. Wkshp (IMW)*, Marseille, France, Nov. 2002.

[3] T. Lang, G. Armitage, P. Branch, and H.-Y. Choo, "A Synthetic Traffic Model for Half Life," in *Australian Telecommunications Networks & Applications Conference (ATNAC)*, Melbourne, Australia, Dec. 2003.

[4] S. Zander and G. Armitage, "A Traffic Model for the Xbox Game Halo 2," in *ACM NOSSDAV*, Stevenson, WA, USA, Jun. 2005.

[5] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "YouTube Traffic Characterization: A View From the Edge," in *In Proceedings of the ACM Internet Measurement Conference (IMC)*, San Diego, CA, USA, Oct. 2007.

[6] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube Network Traffic at a Campus Network - Measurements, Models, and Implications," *Elsevier Computer Networks*, Mar. 2009.

[7] Y. Lu, Y. Zhao, F. Kuipers, and P. V. Mieghem, "Measurement Study of Multi-party Video Conferencing," in *Proceedings of the 9th IFIP TC 6 International Conference on Networking*, 2010, pp. 96–108.

[8] Y. Xu, C. Yu, J. Li, and Y. Liu, "Video Telephony for End-consumers: Measurement Study of Google+, iChat, and Skype," in *ACM Interenet Measurement Conference (IMC)*, Boston, MA, Nov. 2012.

[9] A. Lai and J. Nieh, "On the Performance of Wide-Area Thin-Client Computing," *ACM Transactions on Computer Systems*, May 2006.

[10] K. Packard and J. Gettys, "X Window System Network Performance," in *USENIX Annual Technical Conference*, San Antonio, TX, USA, 2003.

[11] J. Kim, R. A. Baratto, and J. Nieh, "pTHINC: A Thin-Client Architecture for Mobile Wireless Web," in *Proceedings of the World Wide Web Conference (WWW)*, Edinburgh, Scotland, May 2006.

[12] D. D. Winter, P. Simoens, L. Deboosere, F. D. Turck, J. Moreau, B. Dhoedt, and P. Demeester, "A Hybrid Thin-client Protocol for Multimedia Streaming and Interactive Gaming Applications," in *Proceedings of the NOSSDAV Workshop*, Newport, RI, USA, Jun. 2006.

[13] Y.-C. Chang, P.-H. Tseng, K.-T. Chen, and C.-L. Lei, "Understanding the Performance of Thin-Client Gaming," in *Proceedings of IEEE CQR*, May 2011.

[14] M. Claypool, K. Claypool, and F. Damaa, "The Effects of Frame Rate and Resolution on Users Playing First Person Shooter Games," in *ACM/SPIE MMCN Conference*, San Jose, CA, USA, Jan. 2006.

[15] M. Claypool and K. Claypool, "Latency Can Kill: Precision and Deadline in Online Games," in *ACM MMSys*, Scottsdale, AZ, Feb. 2010.

[16] M. Carbone and L. Rizzo, "Dummynet Revisited," *ACM SIGCOMM Computer Communications Review*, vol. 40, no. 2, Apr. 2010.

[17] K. Claypool and M. Claypool, "On Frame Rate and Player Performance in First Person Shooter Games," *ACM/Springer Multimedia Systems Journal (MMSJ)*, 2007.

[18] T. Beigbeder, R. Coughlan, C. Lusher, J. Plunkett, E. Agu, and M. Claypool, "The Effects of Loss and Latency on User Performance in Unreal Tournament 2003," in *NetGames*, Portland, OG, USA, Sep. 2004.

[19] J. Kinicki and M. Claypool, "Traffic Analysis of Avatars in Second Life," in *NOSSDAV*, Braunschweig, Germany, May 2008.