End-to-End and Network-Internal Measurements of Real-Time Traffic to Residential Users

Martin Ellis
School of Computing Science
University of Glasgow
ellis@dcs.gla.ac.uk

Colin Perkins
School of Computing Science
University of Glasgow
csp@csperkins.org

Dimitrios P. Pezaros
School of Computing Science
University of Glasgow
dp@dcs.gla.ac.uk

ABSTRACT

Little performance data currently exists for streaming high-quality Internet video to residential users. Data on streaming performance will provide valuable input to the design of new protocols and applications, such as congestion control and error-correction schemes, and sizing playout buffers in video receivers. This paper presents measurements of streaming real-time UDP traffic to a number of residential users, and discusses the basic characteristics of the data.

Categories and Subject Descriptors

C.2.3 [**Computer Communication Networks**]: Network Operations; C.4 [**Performance of Systems**]: Measurement Techniques

General Terms

Experimentation, Measurement

Keywords

Internet Measurement, Streaming Video, DSL, Cable

1. INTRODUCTION

Streaming video now comprises a significant fraction of Internet traffic [6]. Modern Internet streaming video systems are generally built using either ISP-managed intradomain UDP/IP multicast, or unmanaged interdomain HTTP streaming. Interdomain UDP streaming can potentially offer improved performance compared to HTTP streaming, since it doesn't have to fight TCP dynamics, however traversal of CPE NAT devices has limited its deployability. With the recent publication of ICE [9], NAT traversal has become more manageable and so we revisit the issue of interdomain UDP streaming performance.

In this paper, we present data showing packet level characteristics of synthetic end-to-end UDP traffic transmitted over the open Internet from a well-connected server to residential hosts connected via a number of ISPs, using both ADSL and cable modem connections, in the UK and Finland. We provide an initial summary of the data, highlighting some of the differences between access technologies and ISPs, and due to sending rates and time of day. We further report on TTL-limited probes used to capture hop-by-hop performance characteristics, and on packet-pair capacity estimates of the paths. The data and scripts can be downloaded at `http://csperkins.org/research/adaptive-iptv/`.

These measurements provide insight into the performance of multimedia streaming from the perspective of the home user, and by using TTL-limited probing, we also expose details of the network-internal performance. This insight into the loss and delay characteristics may inform the design of new error-correction schemes and receiver playout buffers.

The remainder of this paper is structured as follows. We outline experimental rationale and methodology in §2 and §3. The format of the data is described in §4. Initial analysis and summary statistics are presented in §5 and §6. We conclude with a discussion of results and related work.

2. RATIONALE

Over-the-top (interdomain) streaming video services generally use HTTP. As a result, they suffer from high latency, due to buffering to compensate for TCP dynamics disrupting packet timing. To achieve low-latency at high quality and data rates, traffic will need to move to non-TCP transport. Accordingly, we study UDP-based interdomain streaming.

We use an active measurement approach, using synthetic RTP traffic [10] running over UDP/IP. This gives us precise control of packet size and timing, allowing us to generate traffic patterns that match commonly used video formats (standard- and high-definition MPEG-2).

We have chosen to implement our measurements using a dedicated platform that can be deployed into residential networks. This platform is built using Soekris net5501 single-board computers running FreeBSD 7 with a custom measurement application. These devices are low-power, easily transported, and can be connected to a home network with zero configuration, providing an environment with known timing behaviour and alleviating the vagaries in performance of home computers administered by unskilled users.

We focus primarily on end-to-end performance, since this is what applications experience, and what drives user perception of the video quality. We also conduct limited hop-by-hop probing, using low-rate TTL-limited packets to solicit ICMP responses from intermediate routers, to attempt to give some insight into the location of loss events, and how timing disruptions evolve across a network path.

MMSys'11, February 23–25, 2011, San Jose, California, USA.

Dates 2009	Link	Rate Mb/s	Time				Len mins
06/27-07/18	*adsl1 8Mb/s*	1	Hourly at :50				1
		2	03:15	10:15	15:15	20:15	10
		4	05:15	12:15	17:15	22:15	10
		6	05:35	12:35	17:35	22:35	10
07/07-07/13	*adsl2 2Mb/s*	1	Hourly at :30				1
		2	04:15	11:15	16:15	21:15	10
06/27-07/04	*cable1 2Mb/s*	1	Hourly at :30				1
		2	04:15	11:15	16:15	21:15	10
07/16-07/22	*cable2 10Mb/s*	1	Hourly at :05				1
		2	04:10	11:10	16:10	21:10	5
		4	04:20	11:20	16:20	21:20	5
		6	04:30	11:30	16:30	21:30	5
		8.5	04:40	11:40	16:40	21:40	5
09/12-09/18	*adsl1 8Mb/s*	1	Hourly at :50				1
		2	03:12	10:12	15:12	20:12	5
		4	03:20	10:20	15:20	20:20	5
		6	03:32	10:32	15:32	20:32	5
09/12-09/18	*adsl3 2Mb/s*	1	Hourly at :05				1
		2	04:12	11:12	16:12	21:12	5
09/22-09/28	*adsl4 8Mb/s*	1	Hourly at :05				1
		2	04:12	11:12	16:12	21:12	5
		4	04:20	11:20	16:20	21:20	5
		6	04:32	11:32	16:32	21:32	5
10/07-10/13	*adsl5 24Mb/s*	1	Hourly at :50				1
		2	03:12	10:12	15:12	20:12	5
		4	03:20	10:20	15:20	20:20	5
		6	03:32	10:32	15:32	20:32	5
10/07-10/13	*adsl6 8Mb/s*	1	Hourly at :05				1
		2	04:12	11:12	16:12	21:12	5
		4	04:20	11:20	16:20	21:20	5
		6	04:32	11:32	16:32	21:32	5

Table 1: Measurement schedule: *dset-A*

Finally, we implement one-way packet-pair probing to estimate the available capacity of the network path. Packet pair has well-known limitations [2], but because of its ease of implementation, it has been widely deployed in some commercial streaming systems. We explore the accuracy of its results on paths where we know the edge link capacity.

3. METHODOLOGY

We describe two datasets, collected between July 2009 and September 2010. The same general methodology was used for each, although specific details evolved over time. Tables 1 and 2 show the residential links hosting receivers, the rates measured, trace schedules, and durations. Link *adsl5* is the same physical link as *adsl1*, but was upgraded by the ISP during the course of the study; the others are distinct links. The server is a well-connected machine at our university.

Measurement traffic is constant bit rate RTP/UDP flows where the RTP sequence number and logical timestamp are augmented with accurate transmission timestamps. Transmission and reception times are logged at the receiver for later analysis. Sender and receiver clocks are synchronised using NTP, allowing us to measure one-way delay variation, but not accurate one-way delay, as discussed in Section 5.

Most of the volunteers hosting our measurement devices

Dates 2010	Link	Rate Mb/s	Time	Len mins
04/25-05/01	*adsl5 24Mb/s*	1	2,5,8,11AM/PM at :22	4
		2	2,5,8,11AM/PM at :28	4
		5	2,5,8,11AM/PM at :34	4
	adsl6 8Mb/s	1	2,5,8,11AM/PM at :40	4
		2	2,5,8,11AM/PM at :46	4
		5	2,5,8,11AM/PM at :52	4
05/13-05/19	*finadsl0 8Mb/s*	1	2,5,8,11AM/PM at :04	4
		2	2,5,8,11AM/PM at :10	4
		5	2,5,8,11AM/PM at :16	4
	cable2 10Mb/s	1	2,5,8,11AM/PM at :22	4
		2	2,5,8,11AM/PM at :28	4
		5	2,5,8,11AM/PM at :34	4
05/25-05/31	*cable3 20Mb/s*	1	2,5,8,11AM/PM at :22	4
		2	2,5,8,11AM/PM at :28	4
		5	2,5,8,11AM/PM at :34	4
06/12-06/18	*fincable0 5Mb/s*	1	2,5,8,11AM/PM at :04	4
		2	2,5,8,11AM/PM at :10	4
		5	2,5,8,11AM/PM at :16	4
	cable4 20Mb/s	1	2,5,8,11AM/PM at :22	4
		2	2,5,8,11AM/PM at :28	4
		5	2,5,8,11AM/PM at :34	4
	cable5 20Mb/s	1	2,5,8,11AM/PM at :40	4
		2	2,5,8,11AM/PM at :46	4
		5	2,5,8,11AM/PM at :52	4
08/01-08/07	*adsl4 8Mb/s*	1	2,5,8,11AM/PM at :22	4
		2	2,5,8,11AM/PM at :28	4
		5	2,5,8,11AM/PM at :34	4
08/28-09/04	*adsl7 8Mb/s*	1	2,5,8,11AM/PM at :22	4
		2	2,5,8,11AM/PM at :28	4
		5	2,5,8,11AM/PM at :34	4

Table 2: Measurement schedule: *dset-B*

Rate (Mb/s)	1	2	4	5	6	8.5
Size (bytes)	1316	1316	1128	1316	752	1128
Spacing (ms)	10	5	2	2	1	1

Table 3: Sending rates, packet sizes and spacings

have monthly-limited or time-of-day-capped bandwidth usage quotas imposed by their ISPs. Extreme connection throttling (to a few kb/s) and excess use fees are possible on exceeding the quota. While we did not consider this in *dset-A*, in *dset-B* we limited the bandwidth consumption of our traces to around 2GB per day for each link, a value that avoids exceeding our volunteers' quotas. Given the video rates we wish to simulate, the total bandwidth consumed per day B_{day} may be calculated as $B_{day} = N \times L \times (B_{1Mb/s} + B_{2Mb/s} + B_{5Mb/s})$, where N is the number of traces per rate per day, and L is trace length.

To give a snapshot of activity at each time, and allow us to capture the variation over different times of day, N was chosen to be 8; this allows L to be as long as 240 seconds. The eight traces per-day capture enough of the diurnal variation seen in the short, hourly traces used previously, and their increased length gives better insight into packet delay distributions. However, we note that even longer traces may also be useful, allowing in-depth time-series analysis of the characteristics within a trace; at the time of writing, such measurements are also being conducted.

Both datasets used a range of transmission rates, chosen

to be representative of both standard-definition and high-definition content. Due to limited scheduling granularity in the measurement system, different packet sizes were required to achieve certain transmission rates (see Table 3). In *dset-A* we used rates chosen to cover the full bandwidth of the links; *dset-B* used a more limited set of rates, matching common MPEG-2 TS packetization rates, that were achievable with fixed packet size. This gives less coverage of the extremes of link capacity, but removes the influence of packet size. Similarly, trace lengths are also standardised in *dset-B*.

TTL-limited hop-by-hop probes and packet pair measurements were taken as part of *dset-B*; *dset-A* is end-to-end only. Logs of which packets were sent with reduced TTL were kept by the sender, along with records of the timing of the corresponding ICMP responses. The TTL-limited packets are sent at a rate of once per second to each of the responsive routers on the path (determined by probing each of the routers on the path before starting the measurement). This low rate was chosen to avoid overloading routers, and to ensure that only one ICMP response was outstanding at any time, to ease matching of response to probe.

The packet-pairs are sent every ten seconds, by generating two packets back-to-back, then leaving a gap of twice the usual interval before the next packet to maintain the average sending rate. The server logs the times both packets were sent, as well as the logical RTP timestamps of each of the packets; these are combined with the arrival timestamps to estimate the path capacity [2].

4. TRACE FORMATS

The datasets are arranged hierarchically, with a directory for each link, and within these, a directory for each rate. The log files are found within these "rate" directories. Reception log files are named according to the time at which they were captured (e.g., `20100501-0222.log` was captured on May 1st 2010, at 02:22am). For each trace, another file shows the anonymised output of a traceroute from the receiver to the sender, taken at the end of the trace. These are named according to the time of capture, with suffix `.rs.traceroute` (e.g., `20100501-0222.rs.traceroute`).

In *dset-B*, the sender also generates log files, named similarly based on the start time of the trace. The file extension represents the type of file (either `.path`, `.pathprobes`, `.packetpairs`, or `.icmp`). Additionally, *dset-B* includes traceroutes from sender to receiver, stored with file extension `.sr.traceroute`.

The format of the packet trace files captured at the receiver and present in both datasets is shown in Figure 1. Each line begins with the capture timestamp (all timestamps measure seconds since 1970). The first line is a header line. The following `(rtp ...)` lines report capture of each RTP packet, giving the decimal values of the RTP header fields [10] with a 1MHz RTP timestamp clock. The `sender_ts` fields is the transmission time inserted by the sender.

Figure 2 shows the format of the additional trace files present in *dset-B* relating to packet-pair and hop-by-hop probing. In particular:

- Before the start of the trace, the sender sends five RTP packets to each hop in turn, checking for multiple IPs per hop, logging the IP addresses of the responses, and timing out if no response is received after one second. Files with the `.path` extension show this mapping from TTLs to (anonymised) router IP addresses; this is used to match the received ICMP messages to the correct TTL-limited packets, as discussed in Section 5.2.
- Files with the `.icmp` extension contain a line for each of the ICMP messages received by the sender within the trace. The 3rd field shows the receive timestamp (seconds since 1970). The 5th field shows the anonymised address of the router that generated the ICMP packet.
- Files with the `.pathprobes` extension contain a line for each of the TTL-limited packets sent within the trace. The 3rd field shows the timestamp (seconds since 1970) just before the TTL-limited packet was sent. The 5th field shows the RTP timestamp ([10], Section 5.1). The 7th field shows the TTL with which the packet was sent. When processing the receiver log file, this log file is consulted to make sure the TTL-limited packets (which stop at the designated router rather than reaching the receiver) are not counted as lost. It is also processed to calculate per-hop loss rates and round-trip times for TTL-limited probes.
- Files with the `.packetpairs` extension contain a line for each of the packet-pairs sent within the trace. The 3rd and 5th fields show the sender and RTP timestamps of the first packet in the pair, and the 7th and 9th fields show the sender and RTP timestamps of the second packet in the pair.

To anonymise the trace files, we process them through a script which replaces IP addresses and hostnames with a token; these have been selected to distinguish, but not identify, the ISPs. The home routers have been named according to the link ID to which they correspond.

5. POST-PROCESSING

This section describes some of the post-processing applied to the traces to extract metrics of interest, including how clock skew is removed from the traces, how one-way delay is calculated, how the logs of TTL-limited packets and received ICMP messages are processed to produce round-trip times, and how the packet-pair measurements are used to estimate capacity. The processed data discussed in this section are also available in the dataset; each of the following sections describe the processing and file formats used. Figure 3 shows an example of these output files.

5.1 Skew Removal / One-way Delay

Conceptually, one-way delay is obtained by simply subtracting send timestamp from receive timestamp; this approach assumes that both clocks are running at the same constant rate, and have zero relative offset. In reality, these assumptions are typically not true, and therefore some external clock synchronisation mechanism is required (as considered in [1]). Although our clients and server are synchronised using NTP, their clocks are still subject to an unknown relative offset β (the difference between the values of the clocks), and relative skew α (the ratio of the rates of the clocks).

End-to-end delays are made up of propagation (fixed), serialisation and queueing (variable) components. The true end-to-end delay of a packet i, d_i (which includes all three components), is the difference between the sender and receiver timestamps (t_i^s and t_i^r) calculated with perfect knowledge of the relative clock offset and skew between sender and

```
dataset-B/adsl5/cbr1.0/20100501-0222.log:
  1272676920.206448 (airmtrecv (version 4.0.1) (build r1000))
  1272676921.297392 (rtp (v 2) (p 0) (x 1) (cc 0) (m 0) (pt 1 unknown) (seq 44954) (ts 0) (ssrc 475832294) (sender_ts 1272676921.276892))
  1272676921.307410 (rtp (v 2) (p 0) (x 1) (cc 0) (m 0) (pt 1 unknown) (seq 44955) (ts 10000) (ssrc 475832294) (sender_ts 1272676921.287114))
  ...
```

Figure 1: Format of Receiver Trace Files

```
dataset-B/adsl5/cbr1.0/20100501-0222.path:
  Hop   1 :   glasgowuni-3    glasgowuni-3    glasgowuni-3    glasgowuni-3    glasgowuni-3
  Hop   2 :   glasgowuni-4    glasgowuni-4    glasgowuni-4    glasgowuni-4    glasgowuni-4
  ...

dataset-B/adsl5/cbr1.0/20100501-0222.icmp:
  icmp  recv_ts 1272676921.340018 icmp_src glasgowuni-3
  icmp  recv_ts 1272676921.408699 icmp_src glasgowuni-4
  ...

dataset-B/adsl5/cbr1.0/20100501-0222.pathprobes:
  pathprobe   send_ts  1272676921.337370 rtp_ts  60000 ttl 1
  pathprobe   send_ts  1272676921.407785 rtp_ts  130000 ttl 2
  ...

dataset-B/adsl5/cbr1.0/20100501-0222.packetpairs:
  packetpair  send_ts1 1272676932.024460 rtp_ts1 10690000 send_ts2 1272676932.024508 rtp_ts2 10700000
  packetpair  send_ts1 1272676942.783357 rtp_ts1 21390000 send_ts2 1272676942.783403 rtp_ts2 21400000
  ...
```

Figure 2: Format of Sender Trace Files (*dset-B* only)

receiver. However, since we have only the measured timestamps $\tilde{t}_i^r$ and $\tilde{t}_i^s$ and don't know the offset or skew, we must use the *measured* end-to-end delay $\tilde{d}_i$.

This $\tilde{d}_i$ is subject to the relative offset (β) and skew (α) between receiver and sender clocks. Since α and β are unknown, they need to estimated from the data; to do this, we followed the approach proposed by Moon *et al.* [8] and implemented by Kohno *et al.* [5], using a linear programming technique to generate estimates for the clock skew and offset, $\hat{\alpha}$ and $\hat{\beta}$. Using these estimates, we were able to correct for skew as shown in (1), producing the *corrected* end-to-end delay $\hat{d}_i$ as an approximation of d_i:

$$\hat{d}_i = \tilde{d}_i - (\hat{\alpha} - 1)\tilde{t}_i^s + \hat{\beta} \quad (1)$$

Assuming the minimum observed delay $\hat{d}_{min}$ corresponds to a packet which experienced minimal queueing delays at the routers along the path, the variation of other packets above $\hat{d}_{min}$ can be seen as a measure of the extent of queueing these packets experienced. We therefore subtract $\hat{d}_{min}$ from the other $\hat{d}_i$ values to approximate queueing delay:

$$DQ_i = \hat{d}_i - \hat{d}_{min} \quad (2)$$

The output of this process is logged in files with the `.qdelay` extension, as shown in Figure 3. The first shows the relative arrival time (in seconds, since the start of the trace); the second shows DQ_i, calculated as described in (2).

5.2 Matching ICMP Responses

The timestamp and target hop of each TTL-limited probe are obtained from the `.pathprobes` file. The timestamp of each received ICMP message is obtained from the `.icmp` file, and the `.path` file is consulted to identify the hop number of the sending router. Using these, each probe is matched to its ICMP response by checking the timestamps of messages received from the router being probed. Since the probes are spaced at 1 second intervals (larger than the highest observed RTT), the ICMP message following a probe is counted as its response, and the RTT is calculated from the send and receive timestamps. We identify losses when a sent probe is not followed by an ICMP response.

The output of this process is logged in files with the `.pathprobe_rtt` extension, as shown in Figure 3. Each line in this file represents a sent probe and ICMP response. The 1st field shows the hop number, and the 2nd and 3rd fields show the send and receive timestamps, respectively. The 4th field contains the RTT for this probe.

```
dataset-B-proc/adsl4/cbr1/20100801-0222.qdelay:
  0.009468 0.00241679
  0.021026 0.00401279
  ...

dataset-B-proc/adsl4/cbr1/20100801-0222.pathprobe_rtt
  1 1280625722.579040 1280625722.580453 0.00141287
  1 1280625723.704560 1280625723.708295 0.00373507
  ...

dataset-B-proc/adsl4/cbr1/20100801-0222.packetpair_dispersion:
  10.747899 0.001209 0.000040 8.303
  21.496965 0.000985 0.000044 10.192
  ...
```

Figure 3: Format of Processed Trace Files

5.3 Calculating Capacity with Packet-Pairs

As described in [2], the estimate of capacity is obtained by dividing packet size (in this case, 1316 bytes) by the arrival dispersion between the packets in the pair.

The output of this processing is contained in files with the `.packetpair_dispersion` extension, as shown in Figure 3. The 1st field shows the arrival time of the second packet in the pair (in seconds since the start of the trace). The 2nd and 3rd fields show the dispersion in seconds between the arrival and departure times, respectively. The 4th field shows the capacity estimate from this pair, in Mb/s.

6. STATISTICS

During the measurement campaign, over 230×10^6 packets were received in roughly 3800 traces while a very low fraction of packets were dropped (0.39%). Reordering was rare, with only 243 packets arriving out of sequence. Our traceroute and ICMP logs indicate the paths we measure are stable, with very few route changes over the measurement period. The following sections describe some of the basic statistics in more detail, including end-to-end packet loss and queueing delay, followed by an analysis of packet-pair and intermediate path measurements.

6.1 End-to-End Packet Loss

Packet loss rates over the traces are typically very low, with many of the traces showing no loss at all. Some vari-

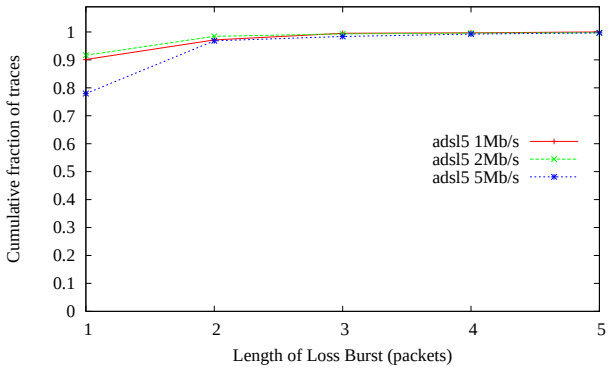

Figure 4: Cumulative Distribution of Loss Bursts (from *dset-B*, link *adsl5*)

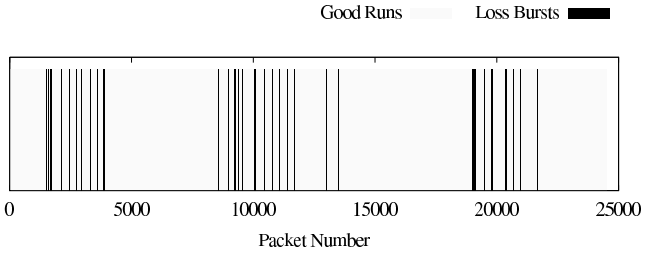

Figure 5: Example of Clustering in Loss Bursts (from *dset-B*, link *adsl5*, 2Mb/s, 2010/04/26, 08:28)

ation is present over times of day, albeit the loss rates are low enough that there is not an obvious trend.

Loss burstiness (i.e., whether they occur closely together in time or randomly spaced out) is more interesting. Figure 4 shows the cumulative distribution of loss burst lengths for *adsl5* (other links are similar). From the CDF, it appears that most bursts consist of a single packet; however, closer inspection reveals that these short loss bursts are clustered together in time. Figure 5 demonstrates the clustering in loss bursts for one trace.

These results show that while overall loss rates are low, the bursty nature of the loss (with groups of bursts clustered together) means that sophisticated error correction and recovery mechanisms may be needed for Internet video systems across these networks.

6.2 End-to-End Queueing Delay

The queueing delay experienced by packets across the dataset varies greatly, between times of day, and between links (with different ISPs showing different behaviours).

Figure 6 shows an example of diurnal variation in queueing delay; half of the links measured (both ADSL and Cable) show this type of behaviour, while the other half show more consistent behaviour over time.

As shown in Figure 7 the queueing delay values tend to cluster around a single mode, with a long right tail (representing large-valued outliers). While only a small proportion of packets show these large delays, their impact may be large since they can disrupt the decoding process.

Previous work [7] found evidence of *heavy-tailed* behaviour in delay on dial-up links, indicating that the values in the tail of the distribution may be extremely large. In order to investigate whether similar behaviour is present for ADSL and Cable links, our ongoing work focuses on tail index estimation and classification.

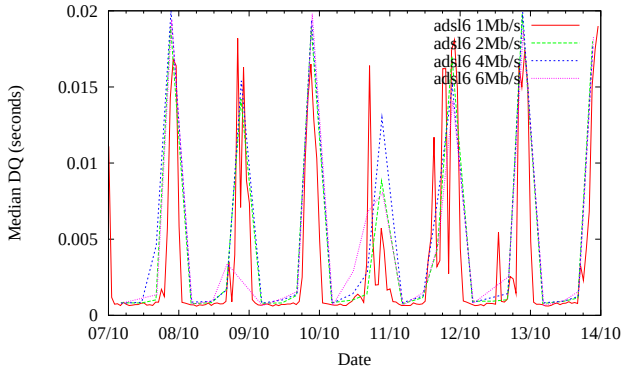

Figure 6: Example of Diurnal Variation in Median Queueing Delay (from *dset-A*, link *adsl6*)

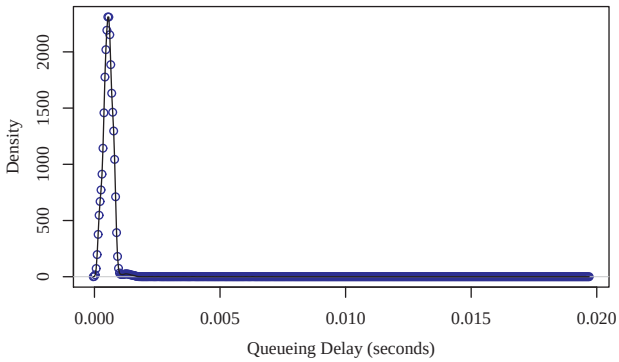

Figure 7: Example of Queueing Delay Distribution (from *dset-A*, link *adsl4*, 4Mb/s, 2009/09/28, 21:20)

6.3 Intermediate Path RTTs

Across the traces, the RTT values produced by each router have a similar clustering; however, there are outliers present, consistent with the right-skewed distribution seen for end-to-end queueing delay. For example, in Figure 8, the distribution of the RTTs can be seen in the boxplot for each hop; routers 7-10 show a significant number of large values. Since the large values are present on these routers, but not on subsequent ones, the variation may be due to the ICMP processing delay. Ongoing work is seeking to determine the effect of each of these components. Using a combined approach, examining these per-hop delay distributions alongside the end-to-end distributions, we hope to model the behaviour of the various parts of the network, and determine the effect of the various parts of the end-to-end path (e.g., core vs. edge networks) on the delay experienced by the receiver.

6.4 Packet-pair Capacity Estimation

Initial investigation of the capacity estimates from the packet-pairs show an interesting difference between ADSL and Cable. Figure 9 shows the distributions of capacity estimates obtained from *adsl6* and *cable2* (each are representative of the other ADSL/Cable links). The ADSL estimate is bimodal (between 10-20Mb/s); this seems consistent with the rated capacity for this link. However, the Cable estimate (around 50Mb/s) is much higher than the speed provided by the ISP (10Mb/s), instead matching the highest rate offered by this Cable ISP. Since Cable uses a time-shared downstream channel, we believe that the estimates of 50Mb/s are

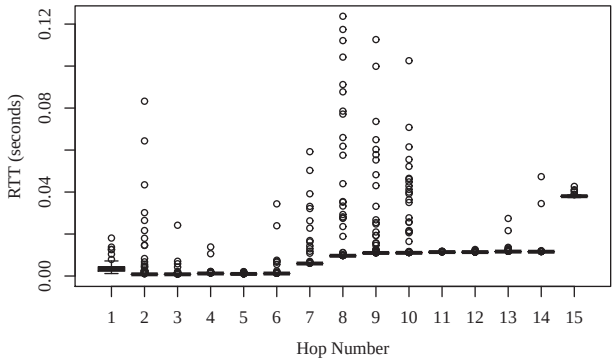

Figure 8: Example of End-to-Middle RTTs (from *dset-B*, link *adsl5*, 1Mb/s, 2010/05/01, 14:22)

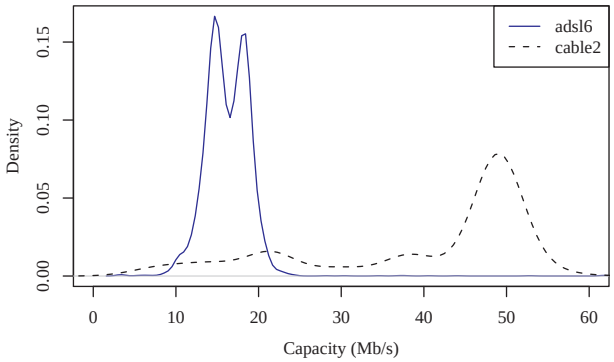

Figure 9: Capacity Estimate Distribution (from *dset-B*, links *adsl6*, *cable2*)

due to both packets in the pair passing through in a single time slice. Further investigation supports this conjecture; using longer trains of packets, we were able to force a pair to be split across a time-slice, producing a lower capacity estimate. Although this technique fails to accurately measure capacity for the Cable links, it might instead be used for link-type classification.

7. RELATED WORK

Prior work has focused on studying peer-to-peer TV [3], backbone performance of IPTV traffic [4], UDP streaming over dial-up links [7] or TCP streaming to broadband users [12]. However, only the data from [4] is publicly available. We publish our measurements of UDP streaming over ADSL and Cable networks, applying similar analyses as [7].

8. DISCUSSION AND CONCLUSIONS

We have presented a dataset of containing measurements of UDP-based streaming to residential Internet users, looking at the loss, reordering, and delay characteristics of these streams. We have also presented intermediate path measurements that aim to correlate end-to-end behaviour and packet-pairs to estimate the capacity of the path.

The clustered bursty loss behaviour we have seen means that simple error correction techniques (such as parity FEC) will be insufficient to mask packet losses. Using retransmissions or more complex FEC techniques will therefore be required to provide acceptable video quality. The queueing delay behaviour we see provides insight into the design of receiver playout buffers for streaming video systems. The distribution of delay does not show tight bounds on the range of expected delays, implying that buffers cannot be optimised in this way. Instead, the receivers will need to cope with occasional highly delayed packets, possibly requiring larger playout buffers.

The differences observed in capacity estimation between packet-pair measurements on ADSL and Cable links pose an interesting question for further work; namely, whether receivers can determine their access type from the incoming video stream (compared with [11], which uses upstream characteristics to distinguish Cable links). This technique would allow the video sender to adapt the stream for each receiver, tuning for their particular access network type.

Future work will include further analysis of the queueing behaviour experienced by the data traffic, using both end-to-end queueing delay and intermediate path RTT measurements. This will inform the design of new protocols and applications for streaming. Development of a link-type classifier based on dispersion of downstream traffic, and effective capacity estimation for these links will also be useful for Internet video systems.

Acknowledgements

This work was supported by Cisco Research and the UK EPSRC. Thanks to Jörg Ott and Alex Koliousis for helpful suggestions, and to the volunteers for hosting measurements.

9. REFERENCES

[1] L. De Vito et al. One-Way Delay Measurement: State of the Art. *IEEE Trans. Instrum. Meas.*, 57(12), 2008.

[2] C. Dovrolis et al. Packet-Dispersion Techniques and a Capacity-Estimation Methodology. *IEEE/ACM Trans. Networking*, 12(6), 2004.

[3] X. Hei et al. Measurement Study of a Large-Scale P2P IPTV System. *IEEE Trans. Multimedia*, 9(8), 2007.

[4] K. Imran et al. Measurements of Multicast Television over IP. In *Proc. IEEE LANMAN*, 2007.

[5] T. Kohno et al. Remote Physical Device Fingerprinting. *IEEE Trans. Dependable Secure Comput.*, 2(2), 2005.

[6] C. Labovitz et al. Internet Inter-Domain Traffic. In *Proc. ACM SIGCOMM*, 2010.

[7] D. Loguinov and H. Radha. End-to-End Internet Video Traffic Dynamics: Statistical Study and Analysis. In *Proc. IEEE INFOCOM*, 2002.

[8] S. B. Moon et al. Estimation and Removal of Clock Skew from Network Delay Measurements. In *Proc. IEEE INFOCOM*, 1999.

[9] J. Rosenberg. ICE: A Protocol for NAT Traversal for Offer/Answer Protocols. RFC 5245, April 2010.

[10] H. Schulzrinne et al. RTP: A Transport Protocol for Real-Time Applications. RFC 3550, July 2003.

[11] W. Wei et al. Clarification of Access Network Types: Ethernet, Wireless LAN, ADSL, Cable Modem or Dialup? *Computer Networks*, 52(17), 2008.

[12] Y. Won et al. Measurement of Download & Play and Streaming IPTV Traffic. *IEEE Comm.*, 46(10), 2008.