We are all treated equal, aren't we? – Flow-level performance as a function of flow size

Muhammad Amir Mehmood[1], Anja Feldmann[2], Steve Uhlig[3] and Walter Willinger[4]
[1]KICS, UET, Lahore, Pakistan, [2]TU Berlin, Germany, [3]QMUL, London, UK, [4]Niksun, Inc., USA
amir.mehmood@kics.edu.pk,anja@inet.tu-berlin.de
steve@eecs.qmul.ac.uk,wwillinger@niksun.com

Abstract—**Recent Internet studies have reported on continued traffic growth, changes in applications usage, and a proliferation in the adoption of high-speed access links. Any adverse impact that these observed trends may have on Internet traffic flows can result in sub par performance, which in turn results in unsatisfactory user experience. To study such adverse impacts, we examine in this paper the flow-level performance of popular applications across a range of size-based flow-classes and applications. We use out-of-sequence packets, retransmissions, throughput, and RTTs as key flow performance metrics. Leveraging data sets collected from two complementary network environments, we compare these metrics for popular applications and for the up/downstream directions. We show that irrespective of the direction, flows are severely impacted by the specifics of the network, e.g., DSL or CDN and application behavior. We also find that, in general, this impact differs markedly across the different flow-classes. In particular, contrary to popular belief, the small flows from all applications, which make up the majority of flows, experience significant retransmissions, while the very large flows, although small in number, experience very limited retransmissions. In terms of application-related performance, we observe that especially when compared to HTTP, apart from large flows, P2P flows suffer from continuously high retransmissions and low throughput. As for the root cause of these retransmissions, we identify the access part of the network as the main culprit and not the network core.**

I. Introduction

The ubiquity of high-speed Internet access and the popularity of content-rich applications have considerably changed the Internet landscape. High capacity access links have enabled users to perform interactive browsing, stream videos, play online games, and share content for social networking, from a wide range of devices. This has caused a fundamental shift in application usage and thus the traffic mix in the network [1], [2].

The availability of high-speed Internet access has also changed user expectations. Streaming high definition movies should be possible without visual impairments. Online gaming users require low latencies. Even standard Web browsing needs latencies in the order of a few round-trip times. Voice applications expect an experience at least equal to the standard phone. Recent studies from Google, Amazon, Yahoo, and Microsoft have demonstrated that few milliseconds difference in web performance impact business value [3].

High-speed Internet access, however, has not made the network less prone to bottlenecks. The popularity of video content such as NETFLIX and YouTube [4] causes significant strain on the network, especially on peering links between ISPs. Aggressive peering strategies used by application providers, visible through the flattening of the Internet hierarchy, i.e., the *middle-mile*, create bottlenecks at peak loads and result in peering disputes among the big players [5], [6]. Furthermore, P2P applications deliver content to users, by relying, e.g., on the often limited ADSL upstream capacity of other users. Understanding how much bottlenecks in the network impact flow performance is crucial for application providers, Internet service providers, and end-users.

In this paper, we show that flows of different sizes from various applications receive different service quality. We devise a methodology that targets *individual* flow performance of different applications using passive traces collected at multiple vantage points. Our work differs from prior research in that we consider the following performance metrics: TCP retransmissions, out-of-sequence packets, throughput, and RTTs based on different classes of flows according to their size, hereafter referred to as *flow-classes*. We use TCP retransmissions as a key performance metric as any retransmission in a flow indicates either packet loss or high latency triggering a timeout event. Our aim is to highlight that flows can experience severe degradations that would be considered unacceptable by today's end-users. Our key takeaways and lessons learned are:

- Packet retransmissions vary significantly across different flow sizes. For example, retransmissions for flows smaller than 128KB range from less than 2% to as much as 40%. Unexpectedly, some large flows do not experience any retransmission or reordering.
- P2P (e.g., BitTorrent/eDonkey) and un-classified traffic dominates in terms of packet retransmissions. The average retransmission rate per flow for this traffic is much higher than for other applications, especially in the upstream direction. This confirms the anecdotal evidence of P2P applications creating bottlenecks by monopolizing the upstream capacity. At the same time, very large P2P flows perform reasonably well without incurring high retransmissions.
- Throughput received by the short flows in DSL environment is better compared to the flows of the same size seen by the CDN. However, large (>8MB) CDN flows exhibit higher throughput.
- P2P (e.g., BitTorrent/eDonkey) and un-classified flows

ISBN 978-3-901882-58-6 © 2014 IFIP

have throughput that is at least an order of magnitude smaller than that of HTTP flows. Large P2P flows on the other hand get a comparable throughput to HTTP.

The remainder of the paper is structured as follows: In Section II we describe our methodology. Section III presents details about the data sets we use. We present our results about retransmissions across flow sizes in Section IV. In Section V, we study the behavior of different applications. We discuss related work in Section VI and summarize our work in Section VII.

II. Methodology

To understand traffic characteristics, it is common to summarize the data at the flow level, e.g., using classical 5-tuple flows based on source and destination IP addresses, port numbers, and protocol. In our context, we need much more detailed information, in particular about TCP retransmissions, throughput, and RTTs. In addition, we want to use an application level detection mechanism that is not purely port based. In the following, we discuss our approach in detail for gathering all the necessary information.

A. Annotated flow summaries

We use the network intrusion detection system Bro [7] as it provides comprehensive analysis capabilities of TCP connections and is able to handle large data sets. In this context, a *TCP connection* refers to a bi-directional TCP communication which starts with the arrival of TCP SYN packet and terminates with FIN/RST packets by either side. Statistics reported by Bro include start time, durations, originator IP and destination IP, originator port and destination port, application protocol, direction, TCP state, additional flags (e.g., to indicate payload data in both directions), payload bytes and packet counts in both directions, as well as a round-trip-time (RTT) sample. The RTT sample is an estimation of the round-trip-time as obtained from the initial TCP hand-shake using a similar methodology as reported in [8]. Payload bytes are accounted as seen in the packets on the wire rather than being estimated from the TCP sequence space numbers.

To determine the application protocol of the connection, we rely on Bro's Dynamic Protocol Detection (DPD) [9] mechanism. Bro includes a wide range of protocol analyzers including HTTP, BitTorrent, eDonkey, FTP, POP3, SMTP, etc. These analyzers detect application protocols by parsing the connections byte stream and matching it to multiple application signatures.

While Bro is capable of detecting a large base of application protocols, we choose seven main categories of applications for our analysis. Our first four categories are HTTP, BitTorrent, eDonkey, and SSL. In addition, we group traffic which is identified by DPD but does not belong to the first four categories together into the group OtherDPD. This includes traffic from protocols such as FTP, POP3, SMTP, and IRC. The well-known category includes traffic on well-known ports. All the remaining traffic is in Un-classified category.

We convert each connection into two half connections, one in each direction, since we want to do separate analysis for both directions. In the following, we refer to these half connections as *flows*. These flows are then annotated with richer meta-information than typical for standard 5-tuples.

B. Flow-classes

It is well-known [10] that Internet flow sizes are consistent with heavy-tailed distributions. Thus, we use logarithmic classes, referred to as *flow-classes*, for binning flows based on their payload bytes. We bin flows into flow-class i, so that for all flows within the flow-class i we have $2^i < payloadbytes \leq 2^{i+1}$ for $i = 0, 1, 2..n$. The largest flow-class also contains flows larger than 2^n. By analyzing flows separately for each size-based class, we gain insight about the relative behavior of flows across flow-classes. We typically start with a flow-class of 1KB and go up to a flow-class of 1GB.

Part of the motivation for looking at flows across different size-based classes is that different flow-classes may be dominated by different types of flows. For example, some types of video objects have median sizes of 265KB, 802KB, and 1743KB. Likewise, many Google products have flows in the range of 4-16KB [11]. Moreover, recent studies about the changing nature of website complexity [12] have shown that overall median web-page sizes for short, medium, and long web pages has grown to 40KB, 122KB, and 286KB respectively.

C. Retransmissions

While TCP is designed to be robust against packet losses and/or reordered packets, flow performance suffers significantly in their presence, see, e.g., the report by Padhye et al. [13]. Indeed, a significant amount of research has focused on making TCP even more robust, e.g., [14], [15], [16], [17], [18]. Therefore, ideally, we want to include both the number of packet losses and the number of reordered packets in our performance metrics. However, distinguishing between them can be challenging [19]. A first approximation is to consider out-of-sequence packets. The next step is to check if they are due to retransmissions and reordering.

One of the complexity of identifying TCP retransmissions from a passive packet level trace is that the trace is collected in the middle of the network and not at any of the endpoints where the TCP state is available. Thus, the challenge is to infer the TCP connection state, including continuous estimation of RTT, and/or congestion window. Moreover, due to routing asymmetry and/or load balancing techniques, not all packets may be seen at the monitor. We again rely on Bro to implement our mechanism for detecting out-of-sequence and retransmitted packets based on ideas by Paxson [20]. The advantage of Bro is that it already tracks per connection state and scales to large data sets [21]. More precisely we classify out-of-sequence packets as follows:

If a packet with the same sequence number as a previously seen packet is observed, it is considered retransmitted. Note, we use the end sequence number of the data within the packet

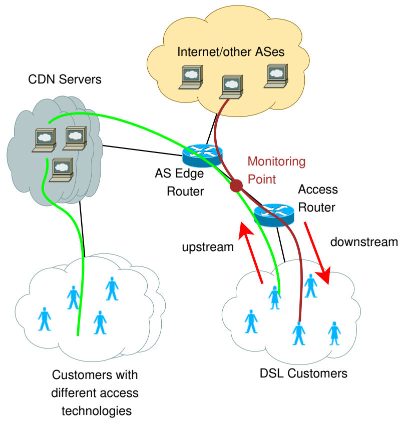


Fig. 1. Measurement setup

for sequence number matching, rather than the start sequence number. This ensures that all cases are excluded where the retransmitted packet transmits more payload than the original packet. Also, we count SYN and FIN retransmissions as they indicate packet losses.

If a packet with an unexpected sequence number according to the ACKs is detected, this indicates a hole in the sequence number space. In this case we compare the timestamp of the out-of-sequence packet to that of the highest sequence number seen so far for this connection. If this time difference is larger than the current minimum RTT estimate of the connection, then the packet is considered retransmitted. The underlying assumption is the expectation that reordered packets are not delayed longer than a round trip time. If the time difference is less than the current minimum RTT estimate and it has a lower IPID, the packet is considered reordered. This strategy allows us to detect out-of-sequence packets and separate them into retransmitted packets and reordered packets.

D. Flow performance metrics

To compare flow performance across different flow-classes, we define four performance metrics. We use throughput, RTT, out-of-sequence, and retransmission packet rate as key flow performance metrics. In the following we present how we compute them. (1) We calculate throughput as the ratio of the payload bytes and the duration of the flow. (2) As round-trip-time estimation, we use the one from the initial three way hand-shake as calculated by Bro. (3) The out-of-sequence packet rate per flow is calculated as the ratio between the sum of the retransmitted and reordered packets, and the total number of packets: $\frac{\sum(retransmit+reorder)}{totalpackets}$. (4) The retransmission rate per flow is calculated as the fraction of retransmitted packets divided by the total packets: $RTX = \frac{retransmit}{totalpackets}$.

III. Data sets and Terminology

In this section, we describe the four data sets used in the paper (see Table I). In this paper, we mainly focus on the MAR-10 DSL trace and the CDN trace. However, we have verified our results across all other traces and will point out differences where applicable. In general, we report our results for the full 24 hours traces as well as for selected hour-long time periods from different times of the day. This allows us to check for time of day effects.

TABLE I
Summary of properties of anonymized traces.

Name	Type	Time	Size	Duration
AUG08	packet (full)	Aug 08	> 4TB	24h
APR09	packet (full)	Apr 09	> 4TB	24h
MAR10	packet (full)	Mar 10	> 4TB	24h
CDN	conn. logs (sampled)	Mar 10	> 50GB	2 weeks

A. Residential broadband ISP traces

Our first data set consists of anonymized packet level traces of residential DSL connections collected at an aggregation router inside a large European ISP. This broadband aggregation router situated in the backbone network of the ISP is a gateway for more than 20,000 DSL customers to the Internet. Data is collected through the help of monitors equipped with Endace cards and is immediately anonymized. The monitor observes traffic from customers with varying line speeds that range from 1200/200 Kbps (downstream/upstream) to 17000/1200 Kbps. In this data set, the distribution of access speed for lines with downstream speeds 1200, 1800, 2300, 3500, 6500, and 17000 Kbps is 15, 2, 20, 23, 31, and 9% respectively. From this monitor, we have collected packet level traces in 2008, 2009, and 2010, each covering a 24 hours period. Overall, the traffic pattern follows a diurnal pattern and does not exceed a link utilization of 45% during the peak hour.

We call traffic that is sent by DSL customers *upstream* and traffic which is received by the DSL customers *downstream*, see Figure 1. Similarly, we refer to the network segment between the DSL customer and the monitoring point as the *local side*, whereas the *remote side* refers to the network segment between the monitor and the rest of the Internet.

Using Bro with DPD, we classify the traffic according to application protocols into seven categories. In our traces we find, consistent with the results of Maier et al. [2], that HTTP is the dominant protocol with a traffic volume share of more than 60% in the downstream direction. In the upstream direction, HTTP has a share of 30%. However, this varies significantly across time and can go as low as 10% during off-peak hours in the early morning. The total traffic from P2P—BitTorrent, e-Donkey—and un-classified is less than 25% of the overall traffic. However, in the upstream direction, the traffic is dominated by P2P (BitTorrent 20%) and un-classified (22%).

B. Content Distribution Network logs

Our second data set consists of connection level logs from the servers of one of the largest content distribution networks (CDNs). We specifically select servers which are serving customers of the same large European ISP from which we gathered the DSL traces. The connections include both those by DSL users as well as all other customers of this ISP, therefore complementing the view from the residential DSL customers. The data within the logs is obtained via kernel level monitoring on the CDN caches and includes low level statistics such as total packets, bytes, retransmitted packets and bytes, RTTs, and durations for each TCP connection. Due to the huge data volume, these logs are only generated for sampled connections.

At the time of when sampling is triggered (i.e., every 1000th packet), the statistics of the sampled connection till that time are recorded and captured on the disk. We note that by definition this process samples connections and is by far the best to get insight for the connections as compared to packet sampling. Packet sampling on the other hand is biased towards the flows with large flow sizes and is prone to miss short flows.

Statistics for all flows are maintained in the kernel, and once the sampling mechanism is triggered, statistics for that flow are recorded to disk. Note, contrary to the previous data, this data is single-sided. It captures only the HTTP traffic in the direction from the CDN to the customer.

IV. Flow size matters

We begin our study by asking the question if out-of-sequence packets are distributed evenly across flow size classes. In this section we focus on out-of-sequence packets as they are a good approximation for retransmissions.

A. Flow size—Motivation

In the past, bulk transfers and their performance have attracted a lot of attention from operators and researchers, for multiple reasons such as optimizing network bandwidth usage [22] to new protocol design [23], [24]. However, while bulk flows contain the majority of the bytes, most of the flows are short [10]. In comparison to the attention that bulk flows have received, short flows have received very little. Yet, it is known that the performance small flows receive can be crucial for the experience of the user. Therefore, some researchers have proposed to give short flows priority over long flows [25]. Still, there is a general belief that short flows do not face as much trouble as long flows since they are not subject to congestion control.

Before delving into the behavior of specific flows bins, we look at the global results. Overall, we observe less than 1.5% out-of-sequence packets across all traces. This is comparable to previous results [26], [11] which observe retransmission rates in the order of 1-2.8%. More precisely, we see 1.2% and 1.5% out-of-sequence packets for MAR10, and CDN.

Out-of-sequence packets are only seen by 9.2% and 16.91% of the connections from MAR10 and CDN. A large number of connections therefore do not see any out-of-sequence packet. This implies that some connections must see more out-of-sequence packets than the average. For example, 1.4% and 3.5% of the connections have more than 20% out-of-sequence packets for MAR10 and CDN. Out-of-sequence packets are therefore not evenly distributed across all connections, which already answers our main question above.

Next, we verify the common belief that short flows do not face much trouble. Figure 2 shows the fraction of out-of-sequence packets. Each plot uses logarithmic flow binning according to flow size (see Section II-B). For each bin, we compute, for all flows within the bin, the percentage of out-of-sequence packets. We then use another binning to show what percentage of flows within a given bin size, have a percentage of out-of-sequence packets that falls within the bin range. This data is then plotted as a stacked barplot with a separate bar per flow size class. Within this bar we show the fraction of flows with an out-of-sequence range larger than 25% at the top and the fraction of flows with no out-of-sequence packets at the bottom. Thus, the y-axis shows, for each flow size bin, the cumulative percentage of flows with a retransmission rate of at least y. In addition, the numbers on the top of the bins indicate the overall percentage of flows within the bin that have out-of-sequence packets.

Figure 2 confirms that the answer to our question is that out-of-sequence packets are not evenly distributed across flow-classes. We will now delve specifically into the different plots of Figure 2.

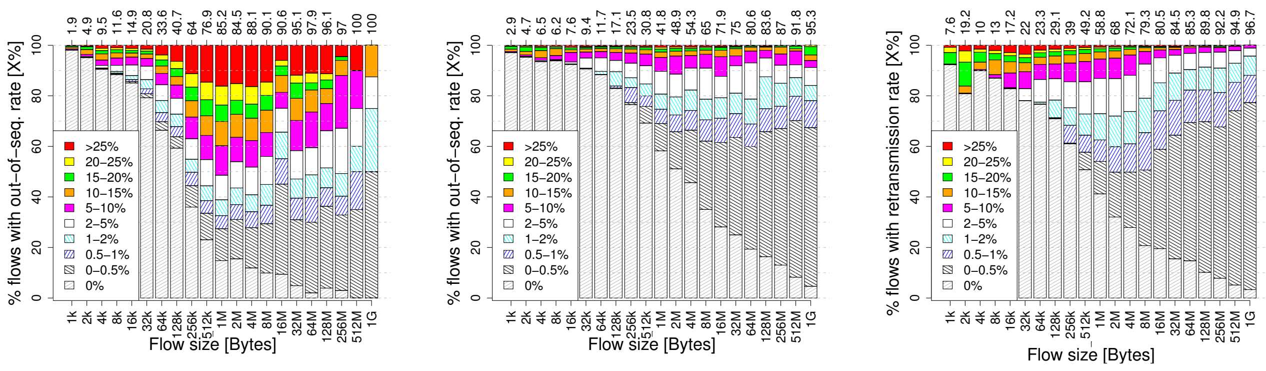
B. Flow size—DSL access

ADSL provides broadband Internet access and typically has highly asymmetric bandwidth at the access. These networks generally rely on an over-subscribed access network, an over-provisioned backbone, and an under-utilized home network. As such, one may expect that small flows manage to sneak through while large flows may suffer from occasional performance problems.

We start with Figure 2(b), which provides the results for the DSL in the downstream direction. As expected, the fraction of flows with no out-of-sequence packets decreases as the flow size increases. Between 2.9 and 17.1% of the flows smaller than 128KB experience at least one out-of-sequence packet during their lifetime. Indeed, the out-of-sequence rate for such flows is relatively low, i.e., in the range of 2 to 15%, as compared to the roughly 4% observed by Jaiswal et al. [19]. We observe similar results in the AUG08 and ARP09 traces (not shown). For example, for the flow-classes 1-128KB, the percentage of flows with out-of-sequence packets varies between 5.6 and 23.3%, and between 3.3 and 20% respectively.

Small flows are not the only ones experiencing very low out-of-sequence rates—some large flows in the tail, i.e., the 1G flow-class, experience no out-of-sequence packets, even though TCP is designed to fully utilize the available network capacity by increasing its network usage until it experiences packet loss. Manual verification has shown that at least some of these are constrained by receive window limitations. About 60% of the flows in the 1M flow-class see an out-of-sequence rate of less than 1%.

Most ISPs DSL offerings provide downstream to upstream ratios of 10:1, which roughly corresponds to the typical data ratios observed when browsing the Web. However, in times of user-generated content, the limited upstream speed can be a major hindrance. Thus, we next check the impact of out-of-sequence packets on the flows in the upstream direction, see Figure 2(a). The overall flow performance, as seen through



(a) MAR10 upstream. (b) MAR10 downstream. (c) CDN.

Fig. 2. Out-of-sequence packet rate per-flow for full trace

out-of-sequence packets, looks vastly different than in the downstream case. First, the percentage of flows with out-of-sequence packets in the 1-128KB flow-classes increases to values between 1.9 up to 40.7%. Our 2008 and 2009 traces show similar values: between 3.3 and 56.1% and between 2.4 and 47.7%, respectively (not shown). More than 40% of the flows of size 256KB-8MB have an out-of-sequence rate above 5%, with around 15% of them experiencing more than 25% out-of-sequence packets. Indeed, the medium sized flows are among those experiencing the largest fraction of very large out-of-sequence packets.

We conclude this subsection with a more nuanced answer to our question: Out-of-sequence packets are not evenly distributed across flow-classes - while small flows can have substantial number of out-of-sequence packets, large flows may have none. Moreover, there are significant differences with respect to traffic direction.

C. *Flow size—CDN's viewpoint*

Next, we focus on a monitoring point close to the servers: the CDN. Note that the CDN data set allows us to analyse per-flow retransmissions, not only out-of-sequence packets. Surprisingly, we again find that 7.6-29.1% of flows smaller than 128KB experience retransmissions, see Figure 2(c). Their number is slightly higher than for DSL (downstream), which can be explained by the diversity of access technologies used by the customers of the CDN[1], including high-speed private networks, cable, mobile, as well as DSL. Overall, we note that the general shape and structure of the plot is similar to the others. The flow behavior as seen from the CDN server perspective is therefore similar to the one observed in the DSL access network environment.

V. Applications and flow-classes

In this section, we ask the question if out-of-sequence rates as well as retransmission rates are evenly distributed across applications. In particular, we want to understand if there are any significant differences between HTTP and P2P either across time or application protocol.

[1]Remember that these are the CDN customers that also belong to the large European ISP.

A. *Application type*

We find that overall the mean percentage of out-of-sequence packets differs significantly by application. BitTorrent, unclassified traffic, as well as eDonkey experience significantly larger out-of-sequence rates. BitTorrent flows see a mean out-of-sequence rate of more than 5%. In particular, BitTorrent sees excessive mean out-of-sequence rates beyond 8% and sometimes even 10% between 5pm and 8pm. Similar observations hold for the upstream direction. However, for the upstream direction, the high out-of-sequence rates occur during the night and sometimes even exceed 12%.

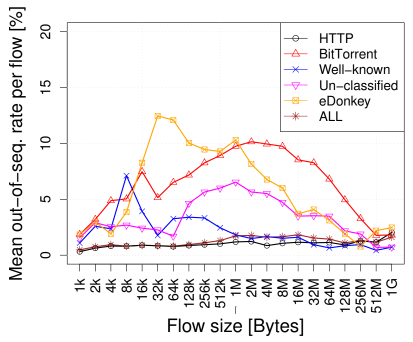
HTTP traffic on the other hand experiences significantly smaller rates of out-of-sequence packets in both directions. In the downstream direction, HTTP flows see a mean rate of 1.1% out-of-sequence packets, slightly lower than the overall average across all traffic. Even lower are the mean out-of-sequence rates for the classes Well-known and SSL. This is a general observation not specific to the MAR10 data set. It also holds for AUG08 and AUG09 in both directions.

The very small out-of-sequence rates for HTTP and well-known, for the DSL upstream traces, stand out. These applications mainly send application requests and TCP ACKs in the upstream direction. Thus, the overall potential of being affected by out-of-order segments is relatively small. It is therefore not surprising to see larger out-of-sequence rates for other applications. The difference between eDonkey and BitTorrent is surprising. Although these are both P2P applications, they seem to impose a different load onto the network, resulting in the different out-of-sequence rates.

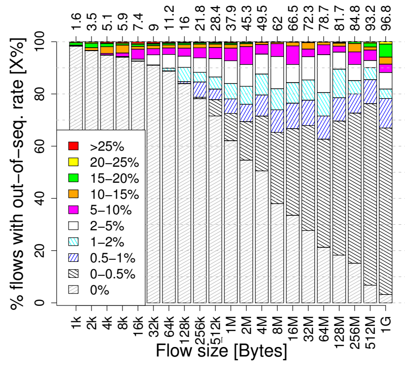
We conclude that out-of-sequence packets are not evenly distributed across time, nor across application types - P2P and un-classified traffic suffer the most.

B. *Flow size—Application type*

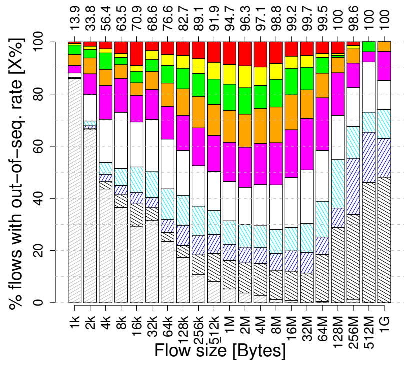
Since BitTorrent flows experience higher out-of-sequence rates compared to HTTP flows, we now take a look at how the out-of-sequence rates for these applications vary across flow sizes. Figures 3(a) and 3(d) show the mean per-flow out-of-sequence rates across flow-classes, by application type. The upstream and the downstream graphs visually look



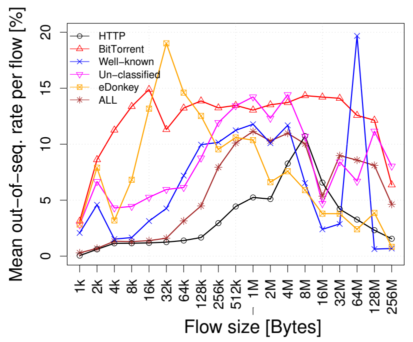
(a) Avg. out-of-seq. rate (MAR10 down).



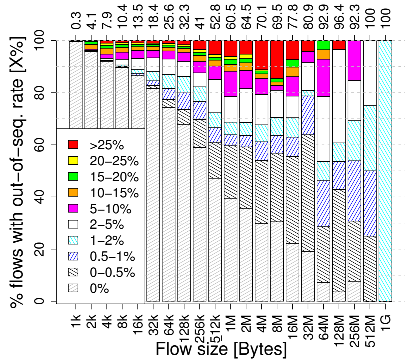
(b) HTTP (MAR10 downstream).



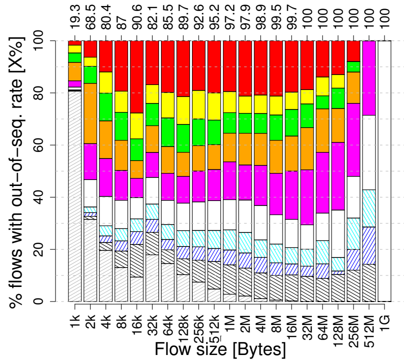
(c) BitTorrent (MAR10 downstream).



(d) Avg. out-of-seq. rate (MAR10 upstream).



(e) HTTP (MAR10 upstream).



(f) BitTorrent (MAR10 upstream).

Fig. 3. Out-of-sequence packet rate per application for 24h duration. (Fig. (c) and (f) use same legend as Fig. (b))

significantly different for some applications and rather similar for others. BitTorrent, eDonkey, and un-classified dominate the top parts of both graphs with high out-of-sequence rates. HTTP and the other applications are at the bottom for the downstream direction and for some parts of the upstream.

We also see that both for eDonkey as well as for BitTorrent, the out-of-sequence rate first increases with flow size and then decreases. The decrease happens earlier for eDonkey, which is why eDonkey overall sees lower out-of-sequence rates. Our explanation is that, as the duration of a flow increases, the likelihood of terminating a large flow, that does not perform well enough, increases with the duration of the flow. Large flows should therefore perform better on average than their smaller counterparts. We will show evidence for this in Section V-D.

We conclude that even at the level of specific applications, out-of-sequence packets are not evenly distributed across flow-classes. On average, larger P2P flows experience fewer out-of-sequence packets than medium sized ones. Moreover, there are again significant differences with respect to traffic direction.

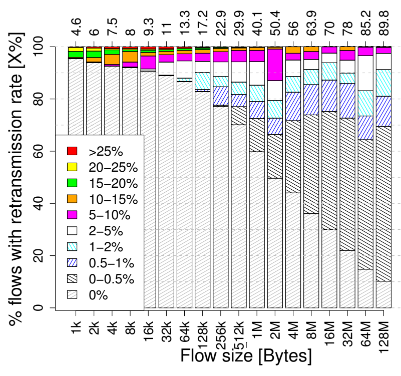
C. *Flow size—HTTP and BitTorrent*

Next, we take a closer look at the ranges of out-of-sequence rates for both HTTP and BitTorrent using the same kind of stacked bar plots as before (e.g., Figure 2). Figure 3 shows plots for HTTP and BitTorrent for the DSL MAR10 data set and both directions. From comparing Figure 2(b) for the whole trace to Figure 3(b) for only HTTP shows little visible difference. We observe that HTTP flows experience smaller out-of-sequences rates than one would expect from the results for the overall trace. Similar observations hold for the upstream direction, i.e., when comparing Figures 3(e) for HTTP and 2(a) for the full trace. However, this time the differences are larger.

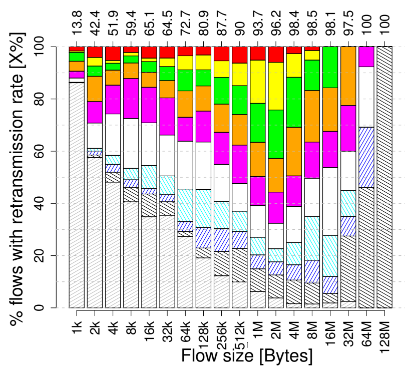
From the comparison of the two upstream directions we find that the out-of-sequence rates are better for HTTP than for the overall—even though worse than for the downstream direction; a small percentage of HTTP flows across most flow sizes (except the smallest and largest) experience very high out-of-sequence rates. One of the reason for higher mean out-of-sequence rates are uploads of user generated content: when users upload relatively large (1MB-16MB) files such as large images or short videos, they are restricted by the limited DSL upstream capacity.

Turning to BitTorrent, we observe from Figure 3(c) and 3(f) that, as for HTTP in the DSL upstream direction, some BitTorrent flows experience severe out-of-sequence rates, across all flow sizes. In the DSL downstream direction, more than 20% of flows from the classes 4KB-128MB experience out-of-sequence rates larger than 5%. In the DSL downstream direction, this is the case for more than 60% of the flows.

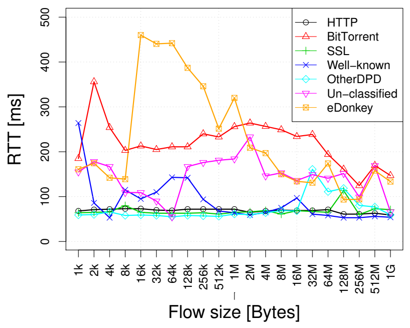
We now return to the comparison of HTTP and BitTorrent. Comparing Figure 3(c) with Figure 3(b) and Figure 3(f) with Figure 3(e), we see major differences, both in terms of overall ranges of out-of-sequence rates as well as different distribution across flow-classes. Part of the reason for this larger fraction



(a) HTTP (MAR10 downstream).



(b) BitTorrent (MAR10 downstream).



(c) RTT across applications.

Fig. 4. Per flow retransmission rate by applications for peak hour and RTT (24h). (Fig. (b) uses same legend as Fig (a)).

of out-of-sequence packets is that P2P applications put more demand on both the uplink as well as the downlink. This can cause congestion either on the DSL link, the home network, or the remote network, leading to packet losses which trigger retransmissions. We have further investigated the burstiness behavior of the flows of different sizes through simulations and we found that mid-sized flows show high burstiness due to aggressive congestion window growth. For detailed results, we refer to our other paper [27].

We conclude that congestion is one of the main factors responsible for the striking difference in terms of out-of-sequence rates for P2P flows and HTTP flows.

D. *Retransmissions—Across flow sizes*

Now, we focus on retransmissions rather than "only" out-of-sequence packets. Our main motivation for focusing on out-of-sequence packets so far is that these are unambiguous. The identification of retransmissions is more challenging due to (a) the multitude of different OSs and different network stacks and (b) the need to estimate some network parameters. Nevertheless, using the methodology outlined in Section II, we can classify which of the out-of-sequence packets are packet retransmissions. Overall, this fraction depends on the trace but is about 92.6%/85.03% for the DSL downstream/upstream environment. This indicates that out-of-sequence packets are for our purposes a good approximation for retransmitted packets. This agrees with the results of Hurtig et al. [28] that estimate that about 5.2% of all out-of-sequence packets are due to reordering.

Figure 4(b) shows the stacked barplot of retransmission rates per flow for only BitTorrent flows in the downstream direction of MAR10. The observation are again consistent with those from Sections V-A–V-C. Moreover, the per flow retransmission rates in the upstream direction are again larger (not shown). Returning to Figures 4(a) and 4(b)—the differences highlight the widely different ways in which TCP is used by various applications.

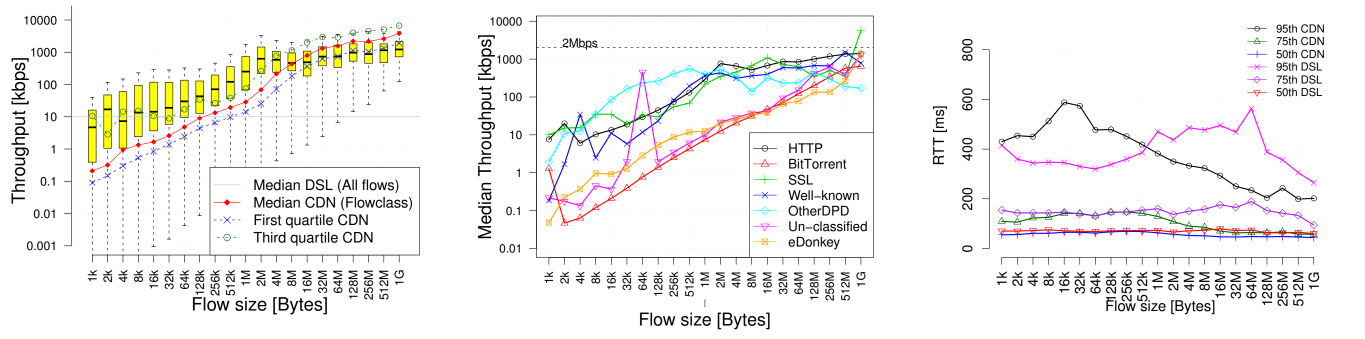
We conclude that the out-of-sequence rate is a reasonable approximation of the retransmission rate. Moreover P2P and un-classified traffic dominate in terms relative number of retransmissions.

E. *Throughput/RTT—Across flow sizes*

One important question we have not yet answered is if large retransmissions, resp., out-of-sequence rates negatively impact flow performance. In principle, if all TCP mechanisms are well utilized, each retransmission should have a negligible performance impact.

Throughput and round-trip-times are among the most relevant metrics to understand TCP flow performance. Throughput can be seen as a measure of raw performance for large flows, as it will measure how well TCP is able to use the available end-to-end capacity. Round trip times on the other hand is very important for short flows, as their limited lifetime requires low RTTs to ensure that the limited amount of data is exchanged within a small enough period of time. In this section, we rely on both metrics to shed light on the individual flow behavior exposed earlier in the paper.

In Figure 5(a), we compare the throughput achieved by different flows for the DSL MAR10 (24h) period and the CDN data sets. The throughput of DSL flows is shown as a boxplot along with the 1st, median, and 3rd quartiles of the CDN flows. Just for comparison, the overall median throughput of DSL flows is 10kbps. We observe that the throughput for small and large flows differ multiple orders of magnitude. When computing summaries of the flow throughput for each flow-class, we find that the performance for flows, e.g., the median, mean, 1st, 3rd quartile, steadily increases roughly exponentially until flow-class 2MB. Then the increase slows down, due to the limited DSL capacity. Compared to the DSL environment, we observe a lower throughput in the CDN data set for short flows. One of the reason is that CDN flows are requested from multiple access technologies, such as mobile, DSL, Cable, that have sometimes limited access capacity. On the other hand, the throughput of large CDN flows (>8MB) show higher median throughput as compare to their DSL counterparts. Indeed, the CDN is optimized to deliver fast large content from close-by servers.



(a) Throughput (y-axis: log scale)

(b) Throughput per application. (y-axis: log scale)

(c) Round-trip-times (RTT)

Fig. 5. Throughput and RTT for DSL (24h) and CDN data across different applications

In Figure 5(c), we present different quantiles of the RTTs across different flow-classes. The CDN flows exhibit very high values of the 95th quantile of the RTTs for short flows. However, this 95th quantile decreases with increasing flow size, consistently with the higher throughput for the large flows of the CDN. We observe that the median RTTs of the CDN are low and similar across flow sizes, though the median for the CDN is slightly lower than for the DSL.

Going back to the observation that retransmissions are not distributed evenly across applications, we ask if the same holds for the per flow throughput. Figure 5(b) plots the median throughput as received by each application protocol by flow size. Our first observation is that throughput varies drastically across application protocols. BitTorrent, eDonkey, and unclassified flows experience by far a worse performance than HTTP flows of the same size. This is a first indication that these flows may suffer from retransmissions.

The low throughput observed for some applications could be due to large RTTs. Figure 4(c) thus plots the median RTT per application across flow-classes. We again see notable differences for P2P and HTTP. The RTTs sampled by HTTP do not differ drastically by flow class. The opposite is observed for P2P. P2P sees increases of the mean RTT well in excess of 200ms. This indicates that either the network distances are significantly larger for P2P traffic than HTTP, or there are significant queues (buffer bloat) in the network and that the queues are contributing to the delay. Verification of the geolocation of the addresses shows that the P2P end-points are a bit further away but not in such a way as to justify an increase in the median RTT from roughly 60ms to more than 200ms. Thus, we conclude that buffering is partially responsible for the RTT increase.

Another observation from Figures 5(b) and 4(c) is that the difference between the applications—in throughput and in RTT— decreases as we consider larger and larger flow sizes. This indicates that the large P2P flows receive reasonable performance which is consistent with our observation that their retransmission rates are lower. We presume that there is some amount of self-selection. Almost all P2P protocols include a mechanism to prefer well performing peers over those that are not, ensuring reasonable performance for the large flows.

We conclude that neither throughput nor RTTs are distributed evenly across flow-classes or application types. Moreover, as the flow size increases, the performance of P2P is approaching that of HTTP (measured in terms of throughput and RTT).

VI. Related work

Much work in the past has tried to better understand Internet properties, for example packet loss [29], [30], [31]. Akella et al. [32] studied Internet bottlenecks, and found that they occur equally within ISPs as well as across peering links. Aikat et al. [33] studied the variability of TCP round-trip times within a connection, and found that the RTT values vary widely. But et al. [34] presented an algorithm to estimate RTT and jitter characteristics of TCP streams monitored at the midpoint of a TCP flow.

Qian et al. [35], in particular, exposed the prevalence of irregular retransmissions across different flow sizes in the Internet. Hurtig et al. [28] have also reported that packet reordering has reduced and is in the order of 5.2% of all out-of-sequence packets. Zhang et al. [36] have found that flow size and flow rate are two highly correlated metrics. The relationship between short flows and application performance has also been studied by Hafsaoui et al. [37].

Lar et al. [14] provide a very comprehensive review of TCP congestion control mechanisms. Siekkinen et al. [38] proposed a TCP toolkit, able to find the primary cause of throughput limitations of TCP flows. Wang et al. [39] studied packet reordering in the Internet and proposed a approach to infer reorder-generating spots in the Internet. Mellia et al. [40] proposed a new heuristic to classify TCP anomalies, including out-of-sequence and duplicate segments.

Recently, Dukkipati et al. [41] has proposed to increase the initial congestion window to 10 to save round trip times for better response times. Similarly, another work from Dukkipati et al. [11] has found that the fast recovery mechanism behaves in a bursty manner and fast recovery should be done using a proportional rate. The behavior of large flows spanning days is discussed in Quan et al. [42]. Similarly, Lee et al. [43] have studied the performance of a congested academic link but have not focused on the short flow performance. They found mean loss rates of 5.77%, consistent with our results.

In the context of applications and losses, Izal et al. [44] have studied the behavior and performance of BitTorrent over a period of multiple months. Pouwelse et al. [45] studied multiple performance aspects of the BitTorrent protocol.

VII. Summary

In this paper, we study the flow-level performance with the help of measurements collected at a large DSL access network and a large CDN provider. Our metrics to gauge flow performance include out-of-sequence packets, retransmissions, throughput, and round-trip times. Our data sets allow us to compare flow performance for the most popular applications (HTTP and P2P) as well as up/downstream directions.

We find that flow performance varies significantly across flow sizes. Small flows, that make up a majority of the flows, experience significant retransmission rates, across all applications. Large flows on the other hand, although fewer in number, can experience limited retransmissions or even none. We observe a marked contrast between HTTP and P2P flows. We highlight the aggressive nature of BitTorrent/eDonkey flows which suffer from continuously high retransmissions due to limited DSL upstream bandwidth that result in low throughput compared to HTTP.

In future work, we will further investigate the relationship between different factors that affect flow performance, such as the interactions between specific applications, the network conditions, and user experience.

References

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *ACM SIGCOMM*, 2010.

[2] G. Maier, A. Feldmann, V. Paxson, and M. Allman, "On dominant characteristics of residential broadband internet traffic," in *ACM IMC*, 2009.

[3] "Strangeloop Site Optimizer – Product brief." [Online]. Available: http://www.strangeloopnetworks.com/assets/Uploads/SO-Datasheet.pdf

[4] "Cisco visual networking index: Forecast and methodology, 2010-2015," http://www.cisco.com/.

[5] T. Leighton, "Improving performance on the internet," *JACM*, 2009.

[6] "Having problems with your Netflix? You can blame Verizon." [Online]. Available: http://goo.gl/NfUvEv

[7] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer Networks*, 1999.

[8] H. Jiang and C. Dovrolis, "Passive estimation of TCP round-trip times," *ACM CCR*, 2002.

[9] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer, "Dynamic application-layer protocol analysis for network intrusion detection," in *USENIX-SS*, 2006.

[10] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: Evidence and possible causes," *IEEE/ACM ToN*, 1997.

[11] N. Dukkipati, M. Mathis, Y. Cheng, and M. Ghobadi, "Proportional rate reduction for TCP," in *ACM IMC*, 2011.

[12] M. Butkiewicz, H. Madhyastha, and V. Sekar, "Understanding website complexity: measurements, metrics, and implications," in *ACM IMC*, 2011.

[13] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," in *ACM SIGCOMM*, 1998.

[14] S. Lar and X. Liao, "An initiative for a classified bibliography on TCP/IP congestion control," *Journal of Network and Computer Applications*, 2012.

[15] E. Blanton and M. Allman, "On making TCP more robust to packet reordering," *ACM CCR*, 2002.

[16] M. Allman, K. Avrachenkov, U. Ayesta, J. Blanton, and P. Hurtig, "Early Retransmit for TCP and SCTP," RFC 5827, 2010.

[17] E. Blanton, M. Allman, K. Fall, and L. Wang, "A Conservative Selective Acknowledgment (SACK)-based Loss Recovery Algorithm for TCP," RFC 3517, 2003.

[18] S. Bohacek, P. Hespanha, J. Lee, C. Lim, and K. Obraczka, "A new TCP for persistent packet reordering," *IEEE/ACM ToN*, 2006.

[19] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, "Measurement and classification of out-of-sequence packets in a tier-1 IP backbone," *IEEE/ACM ToN*, 2007.

[20] V. Paxson, "Measurements and analysis of end-to-end internet dynamics," Ph.D. dissertation, 1998.

[21] H. Dreger, A. Feldmann, V. Paxson, and R. Sommer, "Operational experiences with high-volume network intrusion detection," in *Proc. of ACM CCS*, 2004.

[22] A. Shaikh, J. Rexford, and G. Shin, "Load-sensitive routing of long-lived IP flows," in *ACM SIGCOMM*, 1999.

[23] K. Tan, J. Song, Q. Zhang, and M. Sridharan, "A Compound TCP Approach for High-Speed and Long Distance Networks," in *IEEE INFOCOM*, 2006.

[24] S. Ha, I. Rhee, and L. Xu, "CUBIC: A new TCP-friendly high-speed TCP variant," *ACM SIGOPS*, 2008.

[25] K. Avrachenkov, U. Ayesta, P. Brown, and E. Nyberg, "Differentiation between Short and Long TCP Flows: Predictability of the Response Time," in *IEEE INFOCOM*, 2004.

[26] K. Pentikousis, H. Badr, and A. Andrade, "A comparative study of aggregate TCP retransmission rates," vol. http://arxiv.org/abs/1112.2292, 2011.

[27] M. A. Mehmood, N. Sarrar, S. Uhlig, and A. Feldmann, "Understanding flow performance in the wild," in *IEEE GLOBECOM*, 2013.

[28] P. Hurtig, W. John, and A. Brunstrom, "Recent Trends in TCP Packet-Level Characteristics," in *ICNS*, 2011.

[29] J.-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," in *ACM SIGCOMM*, 1993.

[30] H. Nguyen and P. Thiran, "Network loss inference with second order statistics of end-to-end flows," in *ACM IMC*, 2007.

[31] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-Level Traffic Measurements from the Sprint IP Backbone," *IEEE Network*, vol. 17, pp. 6–16, 2003.

[32] A. Akella, S. Seshan, and A. Shaikh, "An empirical evaluation of wide-area internet bottlenecks," in *ACM IMC*, 2003.

[33] J. Aikat, J. Kaur, F. D. Smith, and K. Jeffay, "Variability in TCP round-trip times," in *ACM IMC*, 2003.

[34] J. But, U. Keller, and G. Armitage, "Passive TCP Stream Estimation of RTT and Jitter Parameters," in *Proc. of LCN*, 2005.

[35] F. Qian, A. Gerber, M. Mao, S. Sen, O. Spatscheck, and W. Willinger, "TCP revisited: A fresh look at TCP in the wild," in *ACM IMC*, 2009.

[36] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the characteristics and origins of internet flow rates," in *ACM SIGCOMM*, 2002.

[37] A. Hafsaoui, D. Collange, and G. Urvoy-Keller, "Revisiting the Performance of Short TCP Transfers," in *NETWORKING*, 2009.

[38] M. Siekkinen, G. Urvoy-Keller, E. Biersack, and D. Collange, "A root cause analysis toolkit for TCP," *Computer Networks*, vol. 52, no. 9, pp. 1846–1858, Jun. 2008.

[39] Y. Wang, G. Lu, and X. Li, "A Study of Internet Packet Reordering," in *Proc of ICOIN*, 2004.

[40] M. Mellia, M. Meo, L. Muscariello, and D. Rossi, "Passive analysis of TCP anomalies," *Computer Networks*, vol. 52, no. 14, pp. 2663–2676, Oct. 2008.

[41] N. Dukkipati, T. Refice, Y. Cheng, J. Chu, T. Herbert, A. Agarwal, A. Jain, and N. Sutin, "An argument for increasing TCP's initial congestion window," *ACM CCR*, 2010.

[42] L. Quan and J. Heidemann, "On the characteristics and reasons of long-lived internet flows," in *ACM IMC*, 2010.

[43] C. Lee, D. K. Lee, Y. Yi, and S. Moon, "Operating a network link at 100%," in *PAM*, 2011, pp. 1–10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1987510.1987511

[44] M. Izal, G. Urvoy-Keller, E. Biersack, P. Felber, A. Hamra, and L. Garcés-Erice, "Dissecting BitTorrent: Five months in a torrent's lifetime," in *PAM*, 2004.

[45] J. Pouwelse, P. Garbacki, D. Epema, and H. Sips, "The BitTorrent P2P file-sharing system: Measurements and analysis," in *IPTPS*, 2005.