A longitudinal analysis of Internet rate limitations

João Taveira Araújo, Raúl Landa, Richard G. Clegg, George Pavlou
University College London
Email: {j.araujo, r.landa, r.clegg, g.pavlou}@ucl.ac.uk

Kensuke Fukuda
National Institute of Informatics
kensuke@nii.ac.jp

***Abstract*—TCP remains the dominant transport protocol for Internet traffic, but the preponderance of its congestion control mechanisms in determining flow throughput is often disputed. This paper analyzes the extent to which network, host and application settings define flow throughput over time and across autonomous systems. Drawing from a longitudinal study spanning five years of passive traces collected from a single transit link, our results show that continuing OS upgrades have reduced the influence of host limitations owing both to windowscale deployment, which by 2011 covered 80% of inbound traffic, and increased socket buffer sizes. On the other hand, we show that for this data set, approximately half of all inbound traffic remains throttled by constraints beyond network capacity, challenging the traditional model of congestion control in TCP traffic as governed primarily by loss and delay.**

I. Introduction

TCP plays a central role in mediating between application needs and network capacity. In the absence of congestion, TCP is responsible for increasing sending rates in a bid to make efficient use of available bandwidth. Conversely, should congestion arise, TCP is expected to reduce its rate. This form of congestion control embedded in TCP is largely credited with performing resource allocation across the Internet and averting congestion collapse. But to what extent does this behaviour describe TCP throughput in practice?

For some types of traffic, throughput is not strictly dictated by the outcome of congestion control. For one, inelastic traffic such as streaming media typically has bounds on the amount of capacity required. Beyond a certain throughput rate, bandwidth probing by TCP is often unnecessary and occasionally harmful. Since TCP drives itself relentlessly towards congestion, real-time applications may suffer from increased latency and jitter. Furthermore, content providers may wish to avoid exceeding the streaming rate for content which may not be consumed in its entirety due to user behaviour, for example channel hopping for video streaming services [19], or client limitations, such as buffering restrictions on mobile devices [20]. Such forms of *application pacing* are often also applied to elastic traffic. In some cases, high throughput is perceived and subsequently marketed as a value added service. One-click hosting services such Rapidshare and Megaupload [6], [22] actively monetise access to large amounts of content both through online advertising and subscription models to bandwidth tiers. Conversely, file sharing applications such as Bittorrent and other peer-to-peer clients allow users to rate limit transfers in order to reduce impact on competing traffic, or to provide incentives for participation [10].

Even beyond such application behaviour, flows may still be constrained by factors not pertaining directly to TCP congestion control. The transport layer is subject to strict bounds within which it can operate, potentially impeded by socket buffer sizes set by OS vendors or tuned by network administrators. There is an upper bound on the window size a receiver can advertise back to the sender; in the absence of windowscale negotiation [7], no TCP connection can exceed a 64KB window. In addition, resource sharing is often subject to local policy. In the absence of adequate methods for readjusting how TCP distributes bandwidth, network operators and system administrators often trade efficiency for predictability, shaping traffic to conform to local notions of fairness or in anticipation for expected demand [15].

Given these different potential sources of rate control, *what can we say about their relative impact on Internet traffic at large*? This paper investigates this question from the point of view of the un-anonymised MAWI dataset [8], that comprises a single transit link over a period of five years (from late 2006 to early 2012). By using routing information external to MAWI, we generate a derived dataset which aggregates flow level metrics across geographic and topological locations. Hence, this paper sheds some light on the asymmetry in global network performance over time and provides a valuable counterpoint to existing US-centric vantage points.

Our main contribution is a re-evaluation of common assumptions regarding Internet flow rates. We achieve this by systematically identifying artificial constraints to TCP traffic throughput across three categories: *application pacing*, *host limiting* and *receiver shaping*. Having reconstructed individual flows, we aggregate these results using an external routing information dataset and relate how each aritificial constraint arises in different contexts and affects different stakeholders over time. This allows us to show not only that flow rates are not typically dictated by TCP congestion control alone, but also that TCP throughput is mostly determined by the actions of the sender. In particular, we show that host limitations have largely been lifted for small flows, with the windowscale option increasing threefold to cover over 80% of all inbound traffic by the end of 2011. Similarly, continuing operating system updates have progressively lifted many of the limitations inherent to socket buffer sizes. These changes have allowed smaller flows to increase throughput at a far higher rate than larger flows, which are more often than not affected by other mechanisms of traffic shaping. This means that, although there is a correlation between flow volume in bytes and throughput, the relationship between the two is nonlinear and has changed with time. Finally, we document the use of receiver shaping on inbound traffic within customer networks as a response to periods of congestion, and show that this technique has been applied mostly on the basis of content characteristics, rather than to high-volume traffic sources.

The paper is structured as follows. We present previous relevant contributions in Section II, and our dataset in Section III. We continue by describing our methodology in Section IV, and some of the implications of our findings in Section V. Finally, we present our conclusions in Section VII.

II. Related Work

This paper builds on a wealth of prior work on understanding Internet traffic and serves as a reappraisal of significant past contributions. Much of the underlying motivation is shared with the landmark study by Zhang et al. [24] on the characteristics of Internet flow rates. Using traces spanning both access, peering and regional links, Zhang et al. analyse traffic according to potential rate limiting factors. Amongst other findings, host window limitations were found to affect over 30% of traffic for the access networks studied. Importantly, the authors found a strong correlation between flow throughput and flow size, postulating that this could derive from user behaviour, with large transfers more likely to be performed over higher bandwidth connections.

Flow characteristics and TCP behaviour at large has since been subject to frequent reassessment. Of particular relevance to the current work are passive studies which delve into the inner mechanisms of TCP. In [13], Jaiswal et al. infer the sender's congestion window by identifying the congestion control variant from the behaviour observed during loss recovery. The use of separate state machines for each variant however proves unscalable given the many flavours of TCP congestion control which have since been deployed. In [17], Lan et al. analyse flows according to size, duration, rate and burstiness and characterise the observed correlations for heavy-hitters specifically, uncovering evidence of increased application influence on flow rates and burstiness and consequently suggest treating flow size and duration as independent dimensions.

In addition to these general investigations, this paper is equally indebted to comprehensive work of a narrower scope. Significant portions of the observed traffic pertain to well known applications which have been previously studied. Rao et al. [20] survey strategies used for video streaming at both Youtube and Netflix and characterise the properties of interleaved *block sending* patterns used to pace streams. These patterns are also the subject of [5], in which the burstiness of Youtube traffic in particular is found to result in considerable losses over residential connections. A large portion of the traffic observed in the MAWI dataset originates from HTTP file sharing services, commonly referred to as one-click hosting websites [6]. In [22], the authors study the characteristics of such traffic over a three month period, detailing the different throttling strategies used by different providers.

Finally, it is important to elucidate what changes in traffic properties are intrinsic to TCP and data transfer, and which ones arise from large-scale changes in the AS-level topology of the Internet. In the decade since publication of [24], the Internet has undergone significant changes, shifting from a broadly hierarchical form to a flatter, more interconnected structure [16], [4]. Given the longitudinal nature of this paper and its focus on interdomain traffic in particular, the insights provided by these studies on the macroscopic effects of content consolidation are discernible within our dataset, and as such are a source of validation for many of the observations herein.

III. Dataset

This section provides an overview of the datasets used in this work and some of the data processing required before approaching the longitudinal study of Internet traffic rate limiting. We use the original, unanonymised traffic traces of the MAWI [8] dataset, a set of daily traces from the WIDE backbone network which provides connectivity to universities and research institutes in Japan. Traffic is captured daily for 15 minutes starting at 14:00JST. Although this dataset extends back largely uninterrupted from late 2001, we focus on just over five years of data following a network upgrade to the monitored link on October 2006.

The monitored link carries mostly trans-Pacific commodity traffic between WIDE customers and non-Japanese commercial networks. We will refer to traffic towards WIDE as *inbound* traffic, whereas traffic originating from within WIDE is referred to as *outbound* traffic.

		TCP data	Traffic (TB)		Count ($\times 10^3$)	
Year	Days	flows (x10^6)	In	Out	AS	Prefixes
2006	91	20.52	0.43	0.45	10.90	56.86
2007	350	102.56	2.11	2.49	17.21	113.79
2008	358	112.26	2.43	2.10	24.74	156.54
2009	364	113.97	2.48	2.53	19.71	143.87
2010	365	113.70	2.58	3.43	20.38	148.03
2011	358	114.74	3.44	5.14	19.99	140.56
Total	1886	5777.55	13.50	16.14	34.12	341.22

TABLE I: Overview of traced MAWI dataset

A preliminary overview of the dataset used is provided in table I. In total, 5.7 billion flows containing data are traced over five largely uninterrupted years; this represents approximately 30 terabytes of TCP traffic. For the purposes of this work, we will focus exclusively on inbound traffic, 60% to 80% of which originates from port 80, referring only to analysis of outbound traffic when providing a wider context for our findings. Given the sender side plays a critical role in shaping traffic, analysing traffic for which the source is restricted to a small set of networks within Japan would be of limited use in accurately depicting traffic trends at large. We instead fix hosts within Japan as sinks, thus sharing a similar perspective on inbound traffic as many other networks.

A. Tracing TCP Metrics

All TCP flows are reassembled and analysed for each daily trace. In addition to the five tuple used to define each connection, we impose two additional restrictions: a contiguous sequence number space and a three minute timeout. These restrictions are helpful to deal with port reuse and unterminated flows respectively. Although the total number of TCP flows increased dramatically in 2011, the number of flows *for which data payload was seen* has remained stable, averaging over 100 million data flows traced per year.

There is much prior work with regards to reconstructing TCP flow from passive measurements and using this information to understand the end-to-end properties of traffic [18], [14], [21], [23]. However, the MAWI traces impose two constraints which require careful consideration, and ultimately led to the use of a custom TCP tracer. The first one is the proportion of bidirectional flows, that is flows, where both forward and reverse path are seen. In the dataset used this fluctuates between 40% and 60% over five years. Most available TCP tracers either ignore or are inadequate at processing flows for which only one direction is observed. The second one is the short duration of each individual trace file. At only 15 minutes of line-rate data capture per day, it is wasteful to ignore flows which are not complete. Although the number of flows for which a SYN and FIN in either direction is observed has remained consistently high until late 2011, these flows are normally *mice*, i.e. flows that tend to be brief and which carry little traffic individually. In contrast, most *elephants* (flows that carry significant traffic individually) have durations that exceed that of each trace file.

Loss is inferred by accounting for *retransmissions* in the upstream data and *out-of-order packets* in downstream data; for the remainder of the paper we will refer to the *end-to-end loss* as the sum of out of order and retransmitted data bytes over the total data bytes in a given direction. Pragmatically, we found this to be an adequate indicator of loss — with the exception of *hanging* TCP connections. In these cases where connectivity is lost, a host will proceed to retransmit packets while performing an exponential backoff. Although this results in negligible overall traffic, it can significantly skew the inferred loss ratio for uncommon destinations for which little traffic exists. To account for these cases, we imposed a 3-second timeout on retransmissions after which we consider the congestion feedback loop to be broken.

Each daily trace in the dataset is processed from a packet level capture into a collection of flow level statistics. This gives us insight into the end-to-end characteristics of traffic. However, since a core objective of this work is to augment this time-based information with data describing the endpoints of each flow, aggregating by location is also required.

B. *Aggregating by Location*

Location information is added by mapping the original source and destination IP addresses to its geographical and topological counterpoints. We use the *routeviews* archives to reconstruct the mapping between each IP and both AS and network prefix; bi-hourly dumps of BGP RIBs are available in the WIDE archives since mid 2003. We reconstruct a daily RIB based on the views provided by contributing ASes, in particular IIJ and APNIC. Since exact routes are not disclosed (there is no record of local policy), we have no knowledge of the route taken by packets; this of course does not hinder our ability to consistently map IPs to ASes. While discrepancies in AS destinations exist between different routeviews contributors, we note that this happens almost exclusively on prefixes for which no actual traffic is seen.

Mapping IP to country is done through the use of GeoLite [3], a commercial geolocation database. While the accuracy of this solution is often disputed, we are not overly concerned with locating traffic at a fine granularity. We will mostly focus on tracking traffic by country and, for larger countries such as the U.S, by region, in order to capture shifts over time. GeoLite proves adequate on both counts. The archive for geolocation data only extends to 2009, before which we must rely on the earliest match. Additionally, we verify if the destination or source AS have maintained the same administrative mapping up until mid 2009 in the relevant RIR (regional internet registry) archives; otherwise, we do not associate a flow to a geographical location. After associating flows to country, region, AS and network prefix for both source and destination IPs, we aggregate flow statistics over each location identifier. This generates a daily collection of location identifiers and associated flow properties, from which we can sketch the geographic and topological properties of the dataset over time.

IV. Flow classification

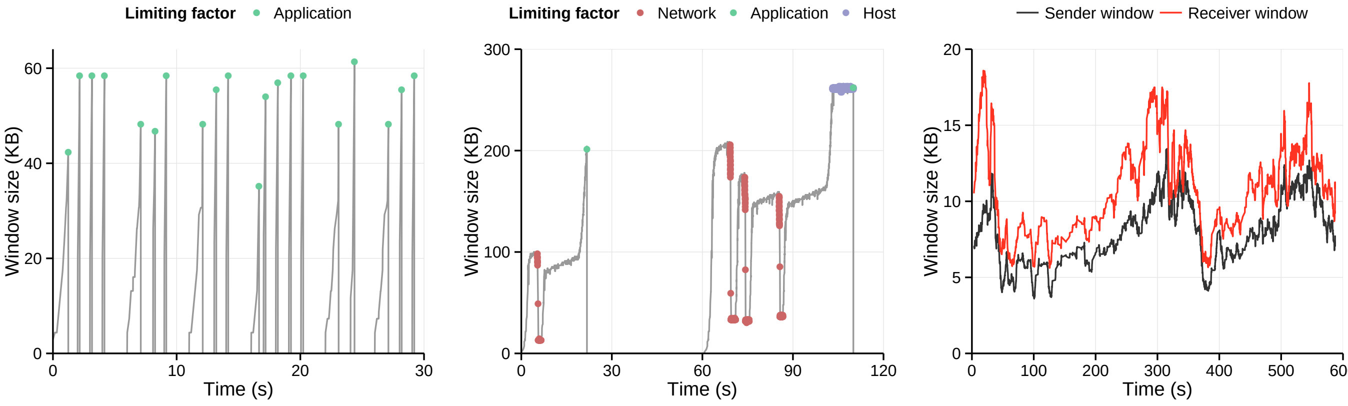
One fundamental precondition to decouple the influence that network loss, host configuration and TCP behaviour has on the throughput experienced by a flow is the reconstruction of the congestion window behaviour of TCP flows on the basis of observed data. Unfortunately, the congestion window value is internal to the sender's TCP state machine and may not manifest itself in the absence of sufficient data from the application layer. A more easily observed quantity which serves as a reasonable proxy for the congestion window is the number of unacknowledged bytes in flight, henceforth referred to as the *flight size*, which can be derived given an accurate estimate of the end-to-end delay. The evolution of both flight size and RTT can in turn be used to ascertain to what extent throughput is regulated by limitations imposed at different layers of the networking stack.

Given a candidate RTT obtained through a similar methodology to the one described in [24], we can aggregate a stream of packets with arrival times $t_1, t_2, \dots$ into a stream of *flights*. Intuitively, a flight is a clustered subset of a TCP flow which exhibits its own temporal coherence; alternatively, it can be thought of as a series of consecutive packets that were (roughly) generated by the sender as a response to the same protocol operation. A flight f_i that begins with the jth packet and ends with the kth is defined to have a *total flight time* $\tau_i = t_{k+1} - t_j$. The algorithmic selection of initial and final packets in such a way that the resulting flights are indicative of TCP behaviour remains an open problem. Since we assume that the RTT provides a natural time frame for the operations of TCP, in the algorithm presented in this work, given an initial packet π_j and an RTT estimate T, the kth (and final) packet is selected to minimise *the flight time error* $e_i = |T - \tau_i|$. This mechanism again follows closely the methodology described in [24], with the exception that we do not attempt to define flights as being both adjacent and disjoint; rather, we decompose flows into a stream of potentially overlapping flights. This helps the algorithm mitigate the deleterious effects of small deviations in the estimated RTT, which alters the properties of each flight. Furthermore, since the flight size is continuous in time, it makes little sense to restrict ourselves to a single sample per round trip time.

Having obtained flight information from each flow, we next consider what is the predominant factor that affects its throughput. Within the context of TCP, we classify flows as being artificially constrained by three distinct processes: *application pacing*, *host limited* and *receiver shaping*.

A. *Application Paced Flows*

A flow whose throughput decreases because it has no outstanding data to send is temporarily limited by the application. Flights can be identified as being *application limited* if



(a) Application paced.

(b) Partially host limited.

(c) Receiver shaped.

Fig. 1: Flight size over time for flows affected by different artificial constraints.

terminated with a packet smaller than the maximum segment size (MSS) and followed by an inter-arrival time greater than the RTT, as consistent with [24]. The underlying reason for this defintion is that most TCP implementations will wait some time for subsequent bytes to be written to the socket if the next packet to be sent is smaller than the MSS, unless the TCP_NODELAY option is set.

Unfortunately, the classification of flights as being application limited is insufficient for our purposes. It can be readily seen that, in practice, all flows for which the final packet is observed contain at least one such flight. For the purposes of our work, we are interested on identifying cases in which throughput is predominantly determined by application behaviour. One such example is illustrated in figure 1a, in which a stream is delivered by periodically writing blocks to the sending socket. The resulting network-level behaviour is distinct from traditional congestion control: short bursts are interspersed with protracted silence. Application limited flights, which terminate on non-MSS packets, are highlighted at the end of each burst.

This motivates our definition of an *application paced* flow, which goes beyond the mere presence of application limited flights and requires that the application play an active role in defining the pace of data exchange. It is important to note that this behaviour (figure 1a) stands in stark contrast to that exhibited when the application simply multiplexes distinct transfers on top of a single transport association (figure 1b). From the perspective of the network, there is little to distinguish the traffic behaviour shown in figure 1b from that arising from independent TCP flows. Application paced connections such as Youtube traffic, however, exhibit a degree of regularity which can potentially be exploited by the network in predicting demand or smoothing bursts.

In order to identify such recurring behaviour, we identify flows as being *application paced* if the period between bursts terminated by *application limited flights* is consistently under 10 seconds and the standard deviation of the intermediate pauses is under one second. This definition is purposely designed to reject flows which exhibit long silence periods due to user interaction, and follows closely the behaviour historically associated with Youtube video streaming [20], [5].

B. Host Limited Flows

Given sufficient bandwidth and traffic to send, a flow may encounter local constraints at either end-host which caps its throughput. For instance, the buffer space allocated on both the sender and receiver side is often pre-configured, and it is common practice to tune these values down on popular servers and managed infrastructure in a bid to conserve memory or bandwidth. A receiver is also limited in the window size it can announce to the remote sender; if the windowscale option [12] is not negotiated during the TCP handshake, the advertised window cannot exceed 64KB.

In both cases, a local decision by either host can determine the upper bound of the flow rate. These *host limited* cases are characterised by a constant window size over time. The methodology described for flight aggregation at the beginning of this section typically generates a large number of flights, representing many likely combinations for a given RTT estimate. In order to identify the flat-lined behaviour of a host limited flow, we first filter the flight stream to remove some of the uncertainty derived from small fluctuations in the RTT. We then select the maximum flight size observed for each RTT interval, and declare a sequence of flights to be host limited if the same maximum was observed over six consecutive RTTs (this is twice the period suggested in [24]). In practice, increasing the period over which the maximum window size is tracked allows us to more accurately discern between host limited behaviour and conservative bandwidth probing, such as that performed during the convex phase of TCP CUBIC [11].

A flow may be host limited for only brief periods of its lifetime, as illustrated in figure 1b. To filter out such cases where host limitations are not the predominant factor in defining flow throughput, we further enforce that in addition to host-limited flights, the average window size must over a flow lifetime be within 10% of the inferred host limit, which is not the case in figure 1b.

In practice, flows can exhibit both application pacing and host limitations, with bursts being sent at a capped window size followed by application pauses. In such cases, a flow will still be classified as being *application paced* if it meets the requirements set out in the previous section, as doing so provides evidence that it controls throughput in spite of the degraded performance provided further down the stack. This line of reasoning applies equally to the occurrence of sporadic loss; so long as block delivery is ensured within the timeframe dictated by the application, it remains in control.

C. Receiver Shaped Flows

A flow which is neither *application paced* or *host limited* can still be artificially constrained by flow control (rather than by congestion control). Traditionally, in TCP the sender is responsible for regulating throughput. However, the receiver can also shape throughput by manipulating the *advertised window* announced on every acknowledgement. Such receiver-window auto-tuning has been available on Windows operating systems since Vista [2], and can also be leveraged by middleboxes in order to throttle inbound traffic [1].

In order to evaluate the potential impact of such behaviour, we propose a correlation-based heuristic to identify receiver-shaped traffic. Figure 1c displays an example of a receiver-shaped connection, in this case throttled by an intermediate middlebox; the correlation is in this case obvious. This is not the case in general, however. Since the advertised window may be fluctuating, it is not always obvious which of the many updates were effectively applied by the sender as successive values supersede each other. Nevertheless, for many flows in which both directions of traffic are observed, it is possible to correlate the evolution of the advertised window with the size of reconstructed flights.

We classify flights as being receiver-shaped if the cross-correlation between the advertised window size and the maximum flight size is statistically significant with a p-value less than 0.05. Harnessing the same packet stream filtering process that we used to detect host limited behaviour, we perform such analysis over a sliding window of 10 RTT intervals. A flow is considered to be predominantly receiver shaped if over half of its flights are flagged as such. We do not perform this analysis on flights which contain out-of-order or retransmitted packets. In these cases, both the receiver and sender window sizes are correlated *by definition*. In the former case, the receiver buffer will temporarily fill expecting the next packet in sequence, in the latter case, TCP will reduce its window.

Unfortunately, the classification of receiver shaped flows can introduce false positives when classifying host limited flows. This happens for flows in which the reverse path was not observed. In these cases, the flow might be receiver shaped in such a manner that the classification heuristic erroneously attributes its behaviour to host limitations. In the absence of additional evidence, this kind of misclassification is difficult to detect explicitly. Instead, we calculate the ratio of receiver shaped flows which would have been incorrectly identified if the reverse path were not observed. This error rate can then be used to evaluate the accuracy of classifier results.

V. Analysis

Having processed each daily trace individually, we proceed by aggregating results longitudinally in order to trace the evolution of constraints affecting TCP across both time and spatial/topological dimensions. We frame our analysis as a revisiting of four commonly held assumptions regarding Internet throughput. Our aim in so doing is to provide much a needed factual verification of these assumptions, which itself can lead to a re-appraisal of Internet throughput modelling efforts. Although all models require simplifying assumptions in the name of analytic tractability, our aim is to inform on which are the best assumptions to make if one is setting out to use or develop an Internet traffic throughput model.

	Limitation (%)				
Year	Application	Host	Receiver	**Total**	Loss (%)
2007	49.47	18.58	0.55	**68.60**	1.29
2008	49.55	17.80	0.69	**68.04**	1.37
2009	47.10	14.50	2.57	**64.17**	1.44
2010	36.78	20.44	3.21	**60.43**	1.22
2011	46.10	13.49	0.60	**60.20**	0.82

TABLE II: Percentage of traffic bytes affected by each constraint by year.

Assumption A. Throughput is primarily shaped by TCP

Internet flow rates are commonly viewed as the output of congestion control embedded at the transport layer. While it is often convenient to model flow throughput according to the steady state behaviour of such algorithms, there are many potential caveats. For one, there is an implicit assumption that the network is the bottleneck. Under such conditions, TCP acts as a distributed optimisation algorithm in allocating capacity to flows. Section IV however presents several cases where such an assumption does not hold. The prevalence of application pacing, host limitations or receiver shaping can all condition the accuracy of models which assume only elastic traffic adjusting to network conditions alone.

Table II displays the extent to which each of these limitations affects inbound traffic in the MAWI dataset over time. The bulk of the volume in bytes is either conditioned by host limits or application pacing. The use of receiver shaping on the other hand is both small in scale and temporally confined to 2009 and 2010. Over five years, the overall effect of the three selected constraints has dropped by close to 10%.

To understand where these dynamics stem from, table III further breaks down these findings by autonomous system, listing the effect of each limitation for the five most significant traffic sources per year. Over the observed five years, traffic remains similarly consolidated: approximately 90% of all inbound traffic is sourced from the top 100 ASes. However, the weight of the most significant sources changes considerably. In 2007 and 2008, a considerable proportion of the traffic exchanged over the interdomain link was content hosted within Japan (NTT, Limelight). From 2009 onwards, most of these local sources established peering connections, bypassing the observed link entirely. This accounts not only for the significant drop of traffic from NTT, but also its altered nature: after 2009 traffic from NTT travelled from further away and was less likely to be application paced.

As the weight of traditional carriers such as Cogent and NTT has waned, ASes known to harbour one-click hosting services such as Choopa, Webazilla, WZ Communications, Carpathia and LeaseWeb have gained significance. Since many websites hosted in these ASes facilitate the distribution of

Year	ASN	AS Name	Traffic (%)	Limitation (%)		
				Application	Host	Receiver
2007	2914	NTT	28.34	65.29	14.68	0.39
	36561	Youtube	15.16	77.41	11.19	0.11
	22822	Limelight	8.12	55.11	21.90	1.37
	15169	Google	3.72	24.11	10.29	0.08
	174	Cogent	2.87	47.65	32.22	0.76
2008	2914	NTT	17.16	62.40	13.71	1.03
	22822	Limelight	10.57	65.40	21.48	0.52
	36561	Youtube	9.15	72.48	12.16	0.09
	2518	BIGLOBE NEC	5.06	84.34	5.84	0.10
	15169	Google	3.52	52.98	17.13	0.18
2009	3462	HiNet	9.93	60.07	4.82	0.05
	15169	Google	8.78	74.79	12.16	0.02
	43515	Google (Youtube)	8.08	83.46	9.83	0.14
	2914	NTT	5.69	39.76	8.37	0.16
	46742	Carpathia (LAX)	4.27	41.04	48.01	2.03
2010	2914	NTT	7.39	21.80	4.91	0.00
	31976	Red Hat	7.03	9.62	41.63	0.00
	7366	Lemuria	5.88	51.95	15.72	5.85
	43515	Google (Youtube)	5.22	77.76	8.41	0.14
	46742	Carpathia (LAX)	4.69	33.06	42.71	4.21
2011	2914	NTT	10.37	50.33	8.19	0.18
	20473	Choopa	8.92	54.03	19.24	0.21
	43515	Google (Youtube)	8.69	69.71	7.56	0.16
	35415	Webazilla	6.05	40.02	11.23	0.95
	40824	WZ Comm.	4.83	42.08	17.43	0.05

TABLE III: AS-Level analysis of throughput limiting.

copyrighted content, they have an incentive to continue using hosted infrastructure rather than deploying their own and risking prosecution. Furthermore, these domains are more likely to host applications which resort to capping the maximum window size as a means of throttling traffic. The increased weight of ASes which resort to these methods, such as Red Hat and Carpathia, significantly contributes to the unexpected increase of host limitations for 2010 displayed in table II.

Overall, we find that flow rates are not typically dictated by TCP congestion control alone. While the impact of application pacing and host limitations on rate control is decreasing, as of early 2012 we find that less than 40% of all inbound traffic had TCP congestion control as the primary rate control mechanism. The reduction of application pacing however may reflect the nature of the observed link, as many traditional streaming providers have migrated towards peering or CDNs, bypassing interdomain links entirely. As such we expect the effect of application pacing to be more pronounced when considering traffic beyond transit. By 2011, successive capacity upgrades have led to a less congested network, but one where predicting how bandwidth is shared is fundamentally harder due to the influence of stakeholders such as content providers and operating system vendors.

Assumption B. Throughput is primarily sender driven

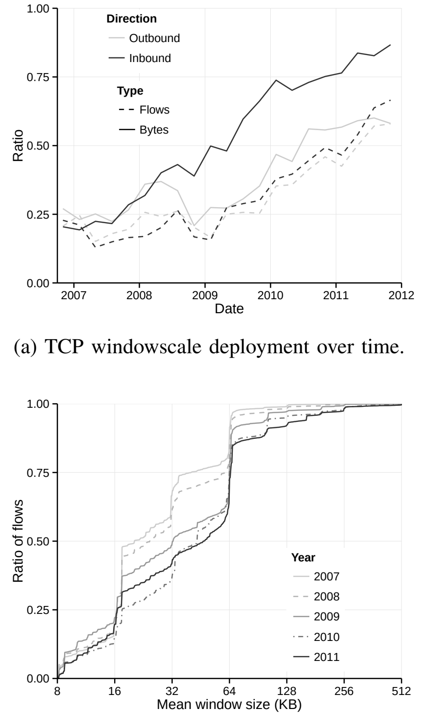
A more widely held and less frequently enunciated assumption is that flow throughput is primarily determined at the sender side. Intuitively, it is in a receiver's best interests to maximize the flow rate, while the sender bears the responsibility for sharing network capacity and reducing the overhead incurred due to losses. The Internet architecture however confers the receiver the ability to throttle rates through flow control. From answering the previous assumption, it is clear that throughput *is* mostly determined by the actions of the sender: receiver shaping and host limitations together affect at most 24% of all traffic. Despite this it is worth understanding the nature of these host limitations in particular, and towards which direction flow control is swinging.

A critical component in determining the upperbound for the congestion window size is the negotiation of the TCP *windowscale* option during the initial handshake. In its absence, a sender cannot have more than 64KB in flight. Furthermore, the *default buffer size* on either end of the connection can also limit the size to which the congestion window can increase. Both settings are primarily subject to operating system configuration. Given that throughput conditions on the receiver side improve as OS upgrades are rolled out, and that the user base within Japan covered by MAWI has remained relatively stable over time, all things being equal we would expect the whole region to exhibit improvements as time progresses. Hence, if we find significant degradation in host limitations, we reason that it most probably does not stem from OS rollbacks or large sets of users with outdated OSes joining the Japanese networks. Instead, we hypothesise that it will be a product of macroscopic shifts in routing or application popularity which lead to a change in *where* traffic originates from.

We test this hypothesis by first verifying windowscale deployment over time. Figure 2a shows the ratio of traffic and flows for which windowscale was successfully negotiated. Results are calculated solely over traffic where the initial handshake was observed. For added context, data for the outbound direction is also displayed. The first result that stands out is the steady increase over time of windowscale usage, rising from 25% of all inbound bytes in early 2007 to almost 80% by late 2011. Furthermore, the effects of content consolidation manifest themselves in the disproportionate coverage of bytes when compared to flows. With the reduced stake of large ISPs in inbound traffic, transit traffic has become dominated by a small set of centrally managed stakeholders such as Google, lowering the effective barrier for deployment of protocol extensions. Conversely, the temporary drop in windowscale adoption for inbound flows in 2009 is due to the increase of traffic from Asian sources, in particular HiNet.

Given the prevalence of windowscaling, the primary source of host limitation should therefore be the configuration of socket buffer sizes. Figure 2b shows the distribution of the average window size for flows which are flagged as being host limited. While the 64KB limit intrinsic to TCP is a common upperbound on window size, other defaults are apparent and have shifted over time. The use of 16KB and 32KB buffer sizes (default buffer sizes for Windows XP and Vista respectively) was progressively phased out over the five year period. In addition to traditional power-of-two increments of the window size, different limits are apparent amongst hosting providers: 50KB, 100KB (The Planet), 160KB (Limelight) and 200KB (SoftLayer), reflecting the overall weight such ASes can have in shaping transit traffic. The influx of Asian traffic in 2009 led to an increase in observed host windows beneath 16KB.

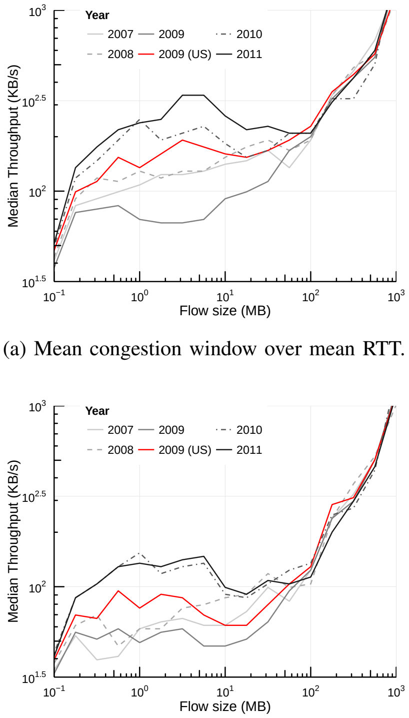
While figure 2b demonstrates that host limitations for inbound traffic have been lifted over time, it still does not adequately answer on what side of the connection they are imposed. Table IV breaks down the proportion of host limited traffic over time for both inbound and outbound direction. In addition to presenting the ratio of flows and bytes affected by host limitations, the relative proportion of traffic identified as being conditioned by the receiver side is also displayed. In



(a) TCP windowscale deployment over time.

(b) CDF of window sizes for host limited flows.

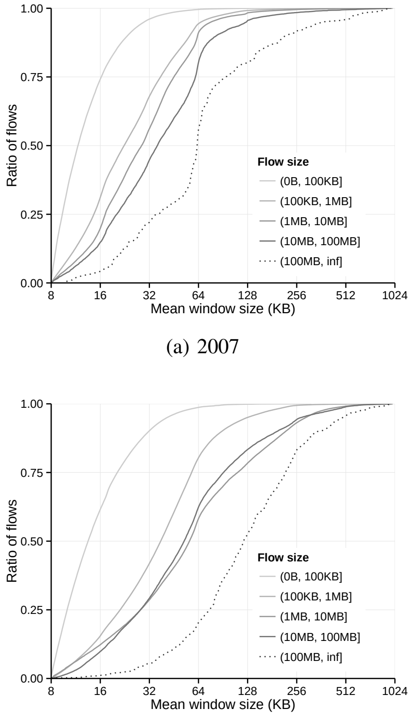
Fig. 2: Longitudinal evolution of TCP window parameters.



(a) Mean congestion window over mean RTT.

(b) Flow size over flow duration.

Fig. 3: Median throughput for inbound traffic by flow size.



(a) 2007

(b) 2011

Fig. 4: CDF of the average window size by flow size by year.

	Total (%)		Receiver (%)	
Year	Flows	Bytes	Flows	Bytes
2007	0.32	18.58	69.01	74.31
2008	0.30	17.80	67.70	71.14
2009	0.27	14.50	60.51	62.80
2010	0.25	20.44	52.34	64.63
2011	0.19	13.49	50.49	60.81

(a) Inbound traffic.

	Total (%)		Receiver (%)	
Year	Flows	Bytes	Flows	Bytes
2007	1.25	22.50	74.62	82.46
2008	1.58	30.21	68.32	84.36
2009	1.25	27.83	60.74	84.20
2010	0.65	24.70	73.75	88.03
2011	0.81	24.23	71.18	85.90

(b) Outbound traffic.

TABLE IV: Percentage of host limited traffic over time by total number of flows and bytes. The proportion for which the receiver side was the bottleneck is also shown.

either direction a very small fraction of flows are affected. Small flows are both numerous and unlikely to last long enough for window limits to be reached or reliably detected. The affected flows therefore tend to be large, and as such can translate into a significant amount of traffic. The proportion of flows and bytes for which the receiver side imposed the maximum window size dropped by 20% and 15% respectively over five years for inbound traffic, reflecting the successive OS upgrades performed for hosts within WIDE. Interestingly, these trends do not surface for outbound traffic: hosts outside Japan were consistently more likely to dictate the maximum window size. In part, this reflects the different nature of the traffic under observation: outbound traffic for this dataset is more geographically and topologically diverse, with content in many cases being retrieved from Japan by residential hosts from within Asia.

The endpoint which ends up dictating the maximum achievable throughput through flow control is typically a function of the OS adoption cycle. With the windowscale option covering 80% of all inbound traffic, the main source of host level constraints are now conservative buffer sizes. For this dataset, hosts internal to WIDE have seemingly been upgraded at a faster rate, or less conservatively, than their remote counterpoints. As such, throughput has become increasingly sender driven over time for inbound traffic.

Assumption C. Throughput is correlated with flow size

Given host limitations are most likely to affect large flows, it's worth considering whether other constraints are applied disproportionately across flow sizes. A commonly held assumption is that throughput is correlated with flow size, which has been verified empirically in previous studies [9], [24]. Much of the data used in these studies however precedes the widespread adoption of high bandwidth connections and use of streaming media, both of which can impact the extent to which contention occurs in the network.

Figure 3 shows the median throughput as a function of flow size, by year. In figure 3a, flow throughput is calculated as the ratio between the mean TCP window size (in bytes) and the mean flight length (in seconds). Compared to the more commonly used ratio of flow size by flow duration, displayed in figure 3b, this method is less susceptible to application behaviour and as such provides a more accurate estimate of the achievable rate. In both cases, flows are binned by size on

	Limitation (%)				
Year	Application	Host	Receiver	**Total**	Loss (%)
2007	14.65	5.45	0.10	**20.20**	1.90
2008	14.99	5.37	0.09	**20.44**	2.27
2009	15.66	3.83	0.55	**20.03**	2.39
2010	10.55	4.18	0.36	**15.09**	2.15
2011	11.13	2.53	0.05	**13.71**	1.19

(a) Flows under 10MB.

	Limitation (%)				
Year	Application	Host	Receiver	**Total**	Loss (%)
2007	61.62	23.07	0.71	**85.40**	0.96
2008	61.49	21.94	0.92	**84.35**	0.88
2009	57.86	17.70	3.28	**78.85**	0.98
2010	43.97	24.45	4.03	**72.45**	0.71
2011	52.95	15.55	0.71	**69.21**	0.62

(b) Flows over 10MB.

TABLE V: Percentage of traffic in bytes affected by each constraint by year, along with aggregate retransmission ratio.

a logarithmic scale, with median throughput calculated across each bin. Due to routing changes and increased congestion, overall throughput in 2009 is lower than other years given there is a greater proportion of traffic from Asian neighbours, particularly over smaller flow sizes. For reference the throughput for traffic from the US alone is plotted for 2009,in which case a more natural yearly progression becomes apparent. For both plots, a clear disparity is visible across flows sizes: for flows in the 10MB to 100MB range, although throughput has consistently increased with time, it has done so at a lower pace than for flows under 10MB.

Our results confirm the notion that the highest throughputs are attained by the largest flows, but they also show that improvements in throughput do not apply equally to all flow sizes. Whereas throughput has consistently improved for low-volume traffic, it has not done so for high-volume traffic. Hence, these findings suggest an increased differentiation between high-value, low-volume traffic whose throughput has markedly increased, and low-value, high-volume traffic whose throughput has stagnated. Although the reported flow sizes are not a reliable predictor of the overall traffic volume amassed over a flow's lifetime given the short time span of each daily trace, this seemingly subverts the notion that flow rates are strongly correlated with flow size in a simple, proportional fashion. This is expounded by further analysing the average window sizes across flow sizes, displayed in Figure 4. In 2007, there is a visible correlation, with larger flows attaining higher window sizes. Furthermore, the distributions cluster prominently around 64KB due to a low rate of windowscale negotiation. By 2011, this clustering is less pronounced, with window sizes increasing across the board, but with larger flows often outpaced by shorter counterparts.

Clearly, the extent to which rates are constrained is closely tied to flow size. In table V we break down the results from table II by flow size. Most limitations will invariably affect larger flows, as applications which shift more traffic, such as streaming media or bulk file transfers, are more likely to either attain or self-impose constraints on flow throughput. Many small flows on the other hand never exit slow start, in which case none of the studied constraints will be reached or readily identified. This dichotomy is reflected on loss rates, which will be higher for flows be regulated by TCP congestion control. Additionally, the discrepancy in loss rates is further exacerbated by geographic properties: traffic exchanged over poor infrastructure tends to be smaller, with flows from China exhibiting particularly high end-to-end loss rates.

			Limitation	(%) of total	
Year	ASN	AS Name	**Receiver**	Bytes	Rexmt
2007	8071	Microsoft	**4.61**	0.69	0.17
	41690	DailyMotion	**2.57**	0.52	0.13
	20940	Akamai	**1.50**	1.49	0.90
	22822	Limelight	**1.37**	8.12	5.01
	19166	ACRONOC	**0.81**	1.40	1.22
2008	21844	The Planet	**2.43**	0.76	0.82
	174	Cogent	**2.02**	1.89	1.39
	19166	ACRONOC	**1.61**	2.02	1.16
	1299	TeliaNet	**1.25**	1.25	1.59
	2914	NTT	**1.03**	17.16	13.24
2009	16276	OVH	**31.97**	1.26	0.29
	3356	Level 3	**6.24**	2.18	1.50
	22822	Limelight	**5.92**	1.60	0.88
	174	Cogent	**5.49**	1.30	0.91
	16265	LeaseWeb	**5.09**	1.98	1.06
2010	1299	TeliaNet	**25.67**	1.31	1.43
	3356	Level 3	**17.33**	2.71	3.16
	16276	OVH	**15.28**	1.84	0.32
	16265	LeaseWeb	**7.70**	3.23	2.16
	29748	Carpathia (Ashburn)	**7.46**	2.62	1.30
2011	7366	Lemuria	**4.45**	1.96	1.56
	46742	Carpathia (LAX)	**1.89**	1.64	0.68
	46179	Mediafire	**1.20**	1.54	0.82
	29748	Carpathia (Ashburn)	**1.08**	2.06	1.27
	35415	Webazilla	**0.95**	6.05	3.33

TABLE VI: AS-Level analysis of throughput limiting.

These results suggest that network upgrades are unlikely to improve performance for significant proportions of traffic. This is most visible in figure 3, where improvements in capacity for coping with higher bursts of activity (figure 3a) has outpaced the actual delivery rate set by applications (figure 3b). Given the popularity of emulating a constant bit rate service over TCP, that no such abstraction is provided at the socket level API is unfortunate.

Assumption D. Throttling primarily affects heavy hitters

We have so far observed that flow throughput is subjugated from TCP by external stakeholders. A third important element in the ensuing tussle is in understanding the role operators can play in imposing their own preferences upon traffic. As such, we now provide a brief overview of the extent, and under what circumstances, customer networks resorted to receiver shaping over the duration of the dataset. From preliminary inspection of table V, it is apparent that receiver shaping was limited in both scope, affecting at most 4% of bytes, and time, being primarily concentrated within 2009 and 2010. A breakdown of receiver shaping by traffic source is provided in table VI, listing for each year the five most affected stakeholders within the top twenty ASes, and their respective contribution to the overall traffic and retransmissions observed yearly.

Prior to 2009, receiver shaping mostly targeted flows which attained the highest throughput, and may have in part been performed by hosts. In addition to affecting Microsoft, which distributes Windows updates using the same flow control mechanisms exploited by middleboxes, some CDN traffic was also reined in. By 2009, and likely as a reaction to increased contention within their networks, the network-level footprint of receiver side throttling shifts from shaping by rate, to shaping by volume. Specific targets start to emerge within OVH, Cogent and Level 3 and TeliaNet, all of which hosted significant file-sharing websites at the time which would continue to be affected throughout 2010. By 2011, receiver shaping mostly subsided, affecting primarily Lemuria (HotFile) and Mediafire. In the presence of increased contention, some customer networks felt obliged to curb traffic which in many cases was already limited by the source. The selected targets of shaping however were neither the biggest contributors in terms of volume, nor the most aggressive senders as reflected by the relative proportion of retransmissions, most likely being singled out instead based on the perceived legality of the content downloaded. When consulting the overall popularity of these targets in table III, it is apparent that some of the most throttled ASes such as Carpathia and Lemuria had been far more popular in previous years, suggesting that users may migrate to other content providers when confronted with lower rates. While successive bandwidth upgrades alleviated the need for throttling, it is unclear whether the middleboxes responsible were discontinued, or limits were merely raised to the extent where the sender side would once again become the bottleneck.

Conventionally, it is assumed that it is in the best interest of content providers to use network resources as fully as possible, whereas ISPs have it in their best interest to police resource usage and control flow rates. However, as shown here, current business practices can create an alignment of incentives where both content providers and ISPs choose to limit the throughput of a particular class of traffic, leading to quality degradation. In these cases, the combination of multiple rate control techniques being applied by different stakeholders may result in a traffic profile whose rate behaves in a manner quite removed from the commonly assumed TCP dynamics.

VI. Acknowledgements

This research was initiated through the NII International Internship Program, and received additional funding from the Seventh Framework Programme of the European Commission, through the FLAMINGO Network of Excellence (318488).

VII. Conclusions

The focus of this work has been on elucidating the main factors that affect flow throughput, but which escape traditional TCP modelling based on end-to-end loss and delay. In particular, we explore the changing role of *host limiting*, *application pacing* and *receiver shaping* in defining flow rates across five years of transit traces. Our results show that for the observed link, over *half* of all inbound TCP traffic can be ascribed to one of the aforementioned constraints. We show that continuing OS upgrades have progressively lifted the artificial throughput constraints imposed by the host stack. In particular, windowscale negotiation for inbound traffic increased threefold in the observed period, covering over 80% of all observed bytes by 2012; in addition, we show that buffer sizes have also shown continuing increases over time.

These developments have significantly improved throughput, in particular for smaller flows. However, we also found evidence of throughput limiting effects independent from available end-to-end capacity. This means that no amount of bandwidth will directly improve TCP rates for a considerable amount of traffic. We show that application-driven techniques for chunked transfer are widely used, accounting for 40% of inbound traffic observed in 2011, and uncover evidence of receiver traffic shaping prior to 2011 based on the modification of the receiver advertised window in a bid to curtail congestion.

References

[1] AppEx IPEQ (IP End-to-End QoS). http://www.appexnetworks.com/white-papers/IPEQ.pdf.

[2] Description of the Receive Window Auto-Tuning feature for HTTP traffic on Windows Vista-based computers. http://support.microsoft.com/kb/947239.

[3] MaxMind GeoLite City. http://www.maxmind.com/app/geolitecity, 2012.

[4] B. Ager, N. Chatzis, A. Feldmann, N. Sarrar, and S. Uhlig. Anatomy of a Large European IXP. *Proc. of ACM SIGCOMM*, 2012.

[5] S. Alcock and R. Nelson. Application flow control in youtube video streams. *ACM SIGCOMM CCR*, 41(2):24–30, 2011.

[6] D. Antoniades, E. P. Markatos, and C. Dovrolis. One-click hosting services: a file-sharing hideout. In *Proc. of ACM SIGCOMM IMC*, 2009.

[7] R. Braden. RFC-1122: Requirements for internet hosts. *Request for Comments*, pages 356–363, 1989.

[8] K. Cho, K. Mitsuya, and A. Kato. Traffic data repository at the WIDE project. In *Proc. of the USENIX Annual Technical Conference*, 2000.

[9] R. G. Clegg, J. T. Araujo, R. Landa, E. Mykoniati, D. Griffin, and M. Rio. On the relationship between fundamental measurements in TCP flows. In *Proc. of IEEE ICC*, 2013.

[10] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, Analysis, and Modeling of BitTorrent-like Systems. In *Proc. of ACM SIGCOMM IMC*, 2005.

[11] S. Ha, I. Rhee, and L. Xu. CUBIC: A new TCP-friendly high-speed TCP variant. *ACM SIGOPS Review*, 42(5):64–74, 2008.

[12] V. Jacobson, R. Braden, and D. Borman. TCP extensions for high performance. RFC1323, 1992.

[13] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. Inferring TCP connection characteristics through passive measurements. *Proc. of IEEE INFOCOM*, 3:1582–1592, 2004.

[14] 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 Transactions on Networking*, 15(1), 2007.

[15] P. Kanuparthy and C. Dovrolis. ShaperProbe: end-to-end detection of ISP traffic shaping using active methods. In *Proc. of ACM SIGCOMM IMC*, pages 473–482, 2011.

[16] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. Internet inter-domain traffic. *ACM SIGCOMM CCR*, 40(4):75–86, 2010.

[17] K. Lan and J. Heidemann. A measurement study of correlations of internet flow characteristics. *Computer Networks*, 50(1):46–62, 2006.

[18] H. S. Martin, A. McGregor, and J. G. Cleary. Analysis of internet delay times. In *Proc. of PAM*, 2000.

[19] F. M. Ramos, F. Song, P. Rodriguez, R. Gibbens, J. Crowcroft, and I. H. White. Constructing an IPTV Workload Model. In *Proc. of ACM SICOMM Poster Session*, 2009.

[20] A. Rao, A. Legout, Y. Lim, D. Towsley, C. Barakat, and W. Dabbous. Network characteristics of video streaming traffic. *Proc. of ACM CoNEXT*, 2011.

[21] S. Rewaskar, J. Kaur, and F. Smith. A performance study of loss detection/recovery in real-world TCP implementations. *Proc. of IEEE ICNP*, 2007.

[22] J. Sanjuàs-Cuxart, P. Barlet-Ros, and J. Solé-Pareta. Measurement based analysis of one-click file hosting services. *Journal of Network and Systems Management*, 20(2):276–301, 2012.

[23] S. Shakkottai, R. Srikant, N. Brownlee, A. Broido, and K. C. Claffy. The RTT distribution of TCP flows in the Internet and its impact on TCP-based flow control. *CAIDA Tech Report*, 2004.

[24] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker. On the characteristics and origins of internet flow rates. *ACM SIGCOMM CCR*, 32(4):322, 2002.