On the Characteristics and Reasons of Long-lived Internet Flows *

Lin Quan
USC/Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Ray, CA 90292
linquan@isi.edu

John Heidemann
USC/Information Sciences Institute
4676 Admiralty Way, Suite 1001
Marina del Ray, CA 90292
johnh@isi.edu

ABSTRACT

Prior studies of Internet traffic have considered traffic at different resolutions and time scales: packets and flows for hours or days, aggregate packet statistics for days or weeks, and hourly trends for months. However, little is known about the *long-term* behavior of *individual flows*. In this paper, we study individual flows (as defined by the 5-tuple of protocol, source and destination IP address and port) over days and weeks. While the vast majority of flows are short, and most bytes are in short flows, we find that about 20% of the overall bytes are carried in flows that last longer than 10 minutes, and flows lasting 100 minutes or longer make up 2% of traffic. We show that long-lived flows are qualitatively different from short flows: they are generally slower, less bursty, and are due to different applications and protocols. We investigate the causes of short- and long-lived flows, and show that the traffic mix varies significantly depending on duration time scale, with computer-to-computer traffic more and more dominating in larger time scales.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network Monitoring*; C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks—*Internet*; C.2.6 [**Computer-Communication Networks**]: Internetworking

General Terms

Measurement

Keywords

Long Duration Flow, Computer-to-computer communication

1. INTRODUCTION

Traffic in the Internet is a complex mix of effects from protocols, routing, traffic engineering, and user behaviors. Understanding traffic is essential to modeling and simulation, traffic engineering and planning [3], router design [1], and better understanding of the Internet [17]. There has been a great deal of study of traffic at the protocol level [19, 20], and at time scales of seconds to hours [4–6, 17, 24], and over longer terms for planning [18, 22]. Yet prior work studies either protocol effects at small time scales (seconds to hours) or aggregate effects at large time scales (hours to weeks), but little attempt to bridge this division and understand protocol effects on long-lived traffic.

This paper explores how users and protocols affect long-lived network traffic. Unlike prior protocol studies, we explore traffic that lasts for multiple hours to days. Unlike prior long-term traffic studies, we explore the causes of traffic patterns at the flow-level in multiple time scales, instead of only trends of aggregate traffic. We use the standard flow definition of the 5-tuple of source and destination IP address and port, plus the protocol number, ended by a timeout [22].

There are several reasons that an understanding of long-lived flows is increasingly important. First, understanding long-lived flows is important for network management. While capacity planning can be done on measures of aggregate traffic, several kinds of on-line traffic control have been proposed: protocol trunking [16], optical trunking [11], lambda switching [2], and low-buffer operation [1]. Understanding the feasibility and impact of these approaches requires flow-level traffic characterization.

Second, a scientific understanding of the Internet must investigate the patterns and causes of long-lived traffic. What are the first-order statistical properties of long-lived flows, and how do they differ from short ones? In addition, short-term studies of network packet data have shown self-similar behavior in time scales of seconds to hours [7, 17], but most such analysis stops as diurnal effects dominate.

Finally, we wish to understand the *causes* of long-term flows. Protocol effects dominate sub-second time scales, and human behavior governs diurnal and weekend effects. Some human-centric traffic is no longer bound by human patience, such as "patient" peer-to-peer file sharing [10], and unattended streaming media, perhaps streaming Internet audio in a store, or automated, Internet-based, Tivo-like

*Lin Quan and John Heidemann are partially supported by the US DHS, contract number NBCHC080035, and Heidemann by the NSF, grant number CNS-0626696. The conclusions of this work are those of the authors and do not necessarily reflect the views of DHS or NSF.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IMC'10, November 1–3, 2010, Melbourne, Australia.
Copyright 2010 ACM 978-1-4503-0057-5/10/11 ...$10.00.

devices such as the Slingbox [14]. Computer-to-computer traffic is growing due to automated control and sensing, online backup, and distributed processing in the cloud and across distributed data centers. We hypothesize that at some point in the future, computer-to-computer traffic will eclipse human-driven traffic, just as data traffic has eclipsed voice.

The contribution of this paper is to begin to answer these questions. To this end, we describe new mechanisms for *multi-time-scale* flow analysis that allows efficient evaluation of network traffic from time scales of minutes to weeks (Section 3). We have operated this system for more than six months, taking data from a regional network. Second, we document the presence of long-lived flows, showing 21% of Internet traffic (by bytes) are carried by flows longer than 10 minutes, and nearly 2% are carried by 100 minutes or longer flows (Section 4.1). Finally, in Section 4.2 we begin to evaluate the causes of such traffic, exploring how protocol mix changes as a function of time scale.

2. RELATED WORK

An important aspect of understanding Internet traffic is packet sizes and protocols. Thompson et al. studied the packet size distribution and protocol mixes in one-day period, and diurnal patterns of aggregate traffic in 7-day period [22]. CAIDA has collected several 90-second traces each day, in a period of ten months, and studied the trends of packet lengths and protocol mixes [18]. We use the common 5-tuple flow definition [22], but we are more interested in flow characteristics and traffic mixes across different time scales.

Characteristics of Internet flows have also been studied extensively. Brownlee et al. studied lifetimes of streams (bidirectional flows) [4, 5]. They found that at least 45% of the streams are *dragonflies* (lasting less than 2 seconds), 98% of the streams last less than 15 minutes and the rest 2% being *tortoises*. Similarly, we find that most of the Internet bytes are carried by the vast majority of short flows, but long flows also account for a considerable fraction of bytes (see 4.1.2). Later work studied flow characteristics systematically, showing the correlations between flow size, duration, rate and burstiness [6]. We adopt the similar ideas from this work, but compare flows behavior as a function of duration.

Because of the large volume of traffic, careful sampling techniques have been used to achieve better processing rates. Researchers from AT&T estimated flow statistics by sampling packet streams and exploiting protocol details [8]. Researchers at UCSD used adaptive sampling algorithms to increase Cisco NetFlow system robustness without compromising accuracy (in case of large volume of traffic) [9]. Zhang et al. studied the distributions and causes of different flow rates [24]. They collected sampled traces from a backbone ISP covering from 1 to 24 hours, and unsampled traces ranging from 30 to 120 minutes. They also studied the correlations between flow rates with size and duration, and gave careful analysis on the causes of different flow rates (such as congestion limited, sender/receiver window limited, and application limited). Our work builds on theirs: we continuously collect unsampled IP packet headers, and systematically study the relations between flow durations and other characteristics. We also provide the ability to investigate multi-time-scale flows for efficient analysis and give preliminary analysis of causes of long-lived flows.

Several other groups have exploited flow characteristics for traffic engineering purposes. Shaikh et al. studied load-sensitive routing, but they adopted a conservative, 10-or-more packet definition of long flows. We study several longer time scales and find interesting implications of the long flows. Trunking (with TCP or optical networks [2, 11, 16]) gathers together groups of flows to achieve throughput benefits. Our work identifies long-duration flows that could be used by trunking. Recent work in low-buffer routing has shown the possibility of using very few router buffers (two magnitudes fewer than current commercial practices), provided that traffic is "sufficiently" smooth [1]. We show that long-duration flows are smoother and could be a good candidate for such optimization.

3. DATA COLLECTION AND ANALYSIS

Network packet trace collection is well understood, but sequential processing becomes challenging as datasets stretch from minutes to months. In this section we review our approach to long-term collection of network flows and multi-time-scale analysis of that data.

3.1 Collection and Anonymization

Our campus network operators provide us anonymized packet headers at the main USC connection to our upstream regional network. We use the LANDER system [12] to process data at USC's HPCC compute cluster [23], coordinating parallel processing of 512MB fixed-length, Endace ERF-format, packet-header traces.

The default LANDER policy anonymizes traffic with keys that rotate at regular intervals. Such a scheme is useful because it insures that any accidental information disclosure in one period does not assist unanonymization in other periods. However, key rotation impedes analysis of flows longer than the rotation period. LANDER therefore re-anonymizes all flows with a common, long-term key. We reduce this greater risk through stricter policy controls: we control access to the long-term data and prohibit those with access to attempt unanonymization.

Although our work builds on packet-header traces, a potential direction for future is to start with NetFlow records as a data source. Another interesting direction is to compare characteristics of long flows at different places of the Internet.

3.2 Multi-time-scale IP Flow Analysis

Given our goal of observing long duration flows, we have four problems: what flows are and what to record for each flow; how to manage streaming data and incremental analysis; how to support analysis at very different time scales, from seconds to weeks or more. We consider each of these next.

We use the standard 5-tuple definition of flows: source and destination port and IP address, plus protocol. We convert LANDER's packet headers into flow records using a slightly modified Argus toolkit [13]. Argus flow records provide: the 5-tuple flow identifier (given above), flow start and finish time, number of packets, and number of bytes in the flow. Flows begin with the first packet with a unique 5-tuple, and continue until a timeout τ (currently set to 60 seconds).

We extend Argus to also capture information about flow burstiness, which is defined as variance of bytes over fixed time period T. We record the number of time periods ob-

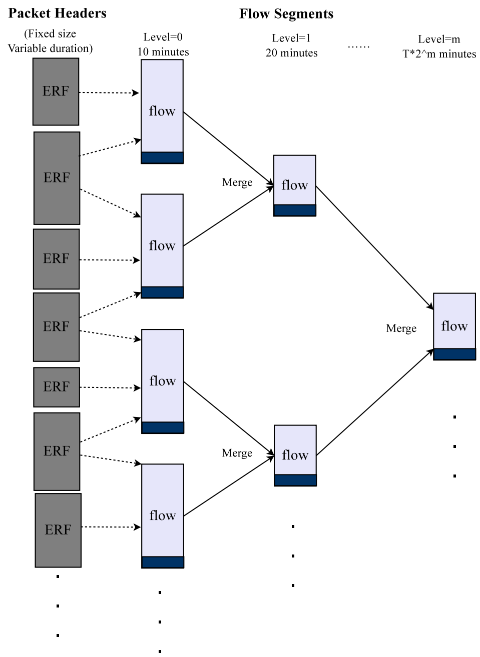

Figure 1: The structure of multi-level flow records: each level has primarily flows with exponentially longer durations, plus a "tail" to permit merging.

served, and the average and square sum of bytes over the time periods. Our base time period for variance is $T = 10$ minutes, the same as our base segment length as described below. This data allows us to compute standard deviation of bytes over T afterwords.

Because we expect to run data collection indefinitely, it is essential that we collect data concurrent with analysis, and that we store data in a manner that supports efficient queries. An easy algorithm would use an in-memory flow table (indexed by the 5-tuple), and update corresponding flow record upon seeing a flow. However, this algorithm can easily run out of memory due to a large number of concurrent flows, particularly with long timeouts. So we divide flow records into *segments* for efficient analysis. LANDER uses fixed-size segments (each 512MB of packet headers, or 1–2 minutes at our current capture rates), and these traces arrive asynchronously, invoking our segment processing engine as it arrives.

We convert these variable-duration segments to *hierarchical, fixed duration* segments to support efficient analysis and queries that span different timescales. We call the intial fixed duration segments *level-0 flow segments*, currently each at a duration of $T = 10$ minutes. When we determine that all packet-header traces needed to cover a flow segment are present, we process them to create the corresponding level-0 flow-segment. Care must be taken because each flow segment typically requires several packet-header traces, and the packet-header trace at the start or end of a flow segment typically span two flow segments. When a trace spans multiple segments, we place the packets corresponding to each segment in seperate flow records in each segment. These records will later be merged into a common flow record in hierarchical merging described next. The left-most column of Figure 1 shows packet headers (dark gray) being converted to level-0 flow-segments.

Each level-0 flow-segment contains 10 minutes of flow records, but long flows will span multiple, possibly hundreds or thousands of segments. Since we cannot sequentially process terabytes of data to make queries for different durations, we assemble level-0 flow-segments into *higher-level segments*. We assemble segments hierarchically in powers of two, so two adjacent level-0 segments are processed to produce one level-1 segment, and so on, with two level-i flows producing a level-$i+1$ flow.

To avoid segments growing in size indefinitely and to allow efficient queries at large timescales, we *prune* the flow contents at each level according to the following rule:

The pruning rule: A level-i segment starting at time t must preserve all flows of duration longer than $T2^{i-2}$ (the duration rule), and all flows that are active in the timeout period the last τ seconds of the trace (the tail rule).

The presence corollary: A level-i segment starting at time t guarantees to contain all flows of durations between $T2^{i-2}$ and $T2^{i-1}$ that start in the time $[t, t + T2^{i-1}]$. It may also contain some shorter flows at the end, and some longer flows (up to $T2^{i}$) which are not complete yet.

(When $i = 0$, the durations start at zero.)

The duration part of the pruning rule keeps each level file small, because each targets a specific time duration and guarantees coverage for that duration. All short flows that are not active at the end of the segment may be discarded. We can prove the presence corollary, because we guarantee coverage for flows that start in the first half of the segment and last for between a quarter and a half of the segment, since by definition those flows must terminate in the segment and are too long to be discarded. We do not guarantee all shorter flows are present, since they will be discarded to keep segment sizes manageable. We cannot guarantee that longer flows are complete since they may stretch into subsequent segments. We show the results of our multi-level storage below in Section 3.4.

The tail part of the pruning rule allows adjacent segments to be merged without loss of information. Only flows that are active in the last τ seconds of the segment are candidates to merge with the next segment, since by our definition of flows they will timeout if the gap is longer. By keeping *all* flows active in this window at the end of the trace we therefore guarantee no information about mergeable flows will be discarded, so we do not accidentally truncate the head of a new long-duration flow. Finally, the rule keeps flows that are *active* in the last τ seconds, more than flows *started* in the last τ seconds—a flow may start anywhere in the segment, and long-running flows will typically span most or all of the segments.

Several details in segment organization support merging and processing. When merging two adjacent level-i segments to create a level-$i+1$ segment, we combine and reorder flow records. We keep flow records sorted by flow start time, so if the level-i files are numbered n and $n+1$, the merge must scan all of file n but only the head of $n+1$. Variance can be combined across segments because we preserve the sum of observations and their squares, not just the computed variance.

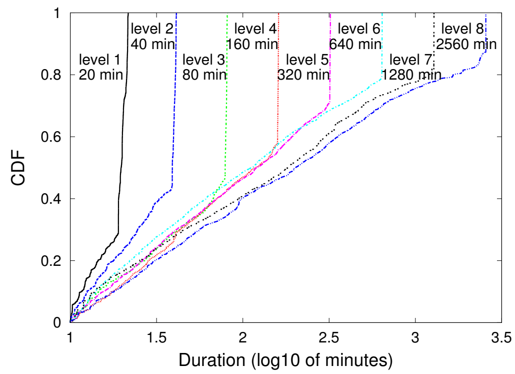

Figure 2: Durations of flows observed at 8 different time scale levels (from 2 days of dataset D1, flows less than 10 minutes truncated).

time scale	all	median	presence
0	4967M	2M	835k
1	652M	570k	7.1k
2	214M	276k	3.1k
3	105M	271k	1.5k
4	53M	274k	949
5	27M	268k	598
6	14M	265k	586
7	7M	300k	301
8	4M	308k	139
9	2.6M	265k	119
10	1M	243k	148
11	846k	846k	71

Figure 3: Number of flows at different time scales: all flows, median per segment, and *presence* flows in one segment (14 days from dataset D8).

Finally, all segment processing is done on a workstation cluster in parallel. Segments are processed and committed atomically (using filesystem rename as the commit method). Concurrent processing of the same file is discouraged by tagging in-process files with a flag, and we recover from crashed processing jobs by timing out flags. We periodically scan the segment tree to catch and correct any missed merges due to races.

3.3 Managing Outages

Very few network tasks can run uninterrupted forever without error—with power outages and scheduled maintenance, continuous operation more than a few months is good. While we tolerate several types of outages, we have experienced multiple gaps, primarily due to software errors in our experimental system. Since May 2009 we have taken 8 traces to date in durations of 8, 9, 15, 23, 40, 65, and 99 days. In the future we plan to bridge brief outages by computing both optimistic and pessimistic flow records around a gap.

3.4 Understanding the Methodology

To illustrate how different time scale flows are stored in different levels of our system, Figure 2 shows the cumulative distribution of flow durations for different levels of segments on a linear-log scale graph. Each line shows a different level segment, starting with level-1 at 20 minutes and doubling at each subsequent level.

Each level shows a range of flow durations. Because of the tail rule, all segments have some very short flows. Because there are relatively few very long flows, the size of high-level segments is dominated by shorter flows. Although each segments at level-i contain flows from zero to $T2^i$ in duration (some of them may not be complete yet), many short flows have been pruned away for clearer view of the longer flows.

In addition, each segment has a large number of flows near the segment duration limit. For example, 70% of level-1 flows are about 20 minutes long, and 57% of level-2 flows are 40 minutes long. These durations indicate flows that last the entire segment and are part of flows that span multiple segments. Their correct duration can only be identified at higher-levels.

To show the advantage of our multi-time-scale storage, Figure 3 shows the number of flows across all files at each level, the median for each level, and how many are valid by the presence rule. We see the number of valid, *presence* flows (bottom line) per segment drops quickly—the true number of long flows is small. The median number of flows plateaus around 300k per segement because segment size is limited by the tail rule and all flows active in the last τ seconds. Finally, the storage requirements (top line) drop exponentially, although they are again limited by the tail rule. We conclude that multi-scale storage is important to study long duration flows.

4. RESULTS

We next describe the results of our measurement system: how do long flows differ from short flows in their characteristics and causes? Since May 2009 we have colected 8 traces. For this paper, we focus on *D1*, a 2-day subset of 15-day capture starting 27 May 2009, and *D8*, a 14-day subset of a 65 day capture starting 19 Feb 2010.

4.1 Characteristics of Long Flows

We first compare flow characteristics: rate, size in bytes, and burstiness as a function of flow duration. Our goal is to understand what long flows are like, and how they differ from short flows. We therefore graph density plots, with darker shades indicating more flows. To quantify distributions at each time scale, we overlay box plots for each time scale, showing quartiles, minimum, and maximum.

Most graphs in this section are generated with *time-scale sampling*: we take one level-i segment for each level ($i \in [1, 11]$, omitting level 0), getting a representative sample from a fraction of the data (Section 3.4). We then select subset of that segment that we can guarantee full capture (flows with duration in $[T2^{i-2}, T2^{i-1}]$) and plot only those flows, discarding the rest. This approach implies that one can compare frequency of some characteristic across a given time scale (for a fixed x value). However, at different time scales (varying x), the absolute number of shorter duration flows are underrepresented relative to longer duration flows.

Figure 4 shows this difference: the left graph uses *both* level-0 segments and one level-1 segment (all flows), while the right uses only one of each level (sampled), so the left has higher absolute densities indicating more flows. Although time-scale sampling under-estimates the total num-

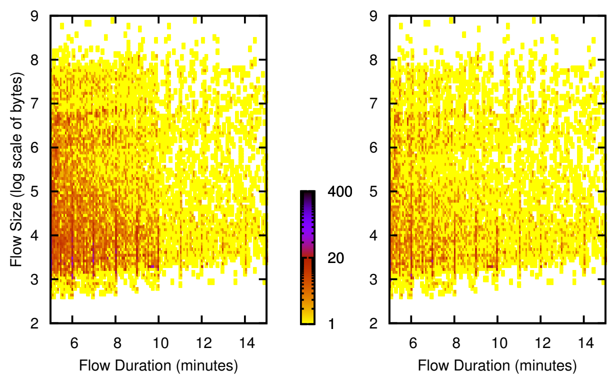

Figure 4: Density plot comparing all (left) and sampled flows (right), duration vs. size (from D8).

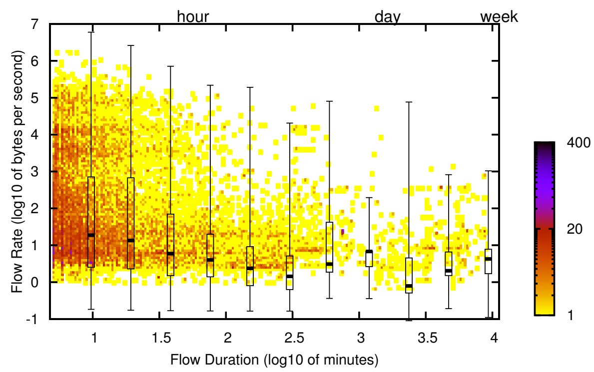

Figure 5: Density plot (log-scale) with quartile boxes of flow duration vs. rate (sampled from D8).

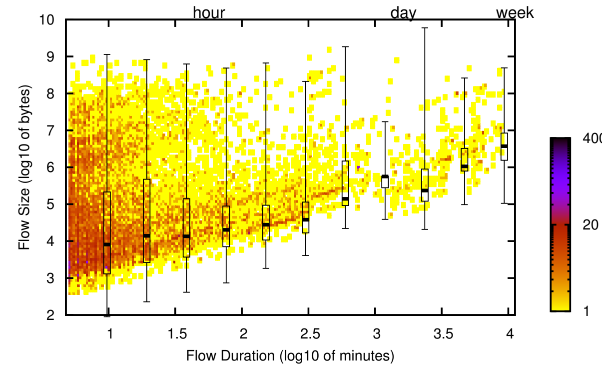

Figure 6: Density plot (log-scale) of flow duration vs. size (bytes) (sampled from D8).

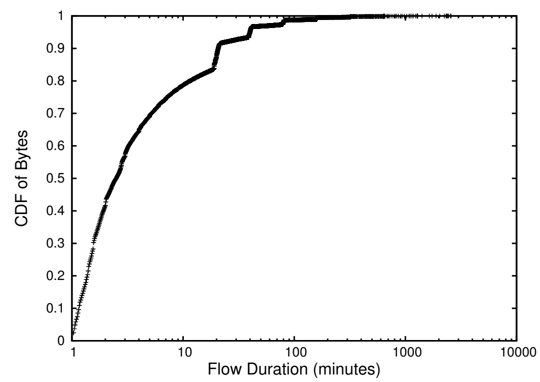

Figure 7: Cumulative distribution of flow sizes (in bytes) of all flows of two days from D1.

ber of flows in the sampled case, it correctly reports the overall trend of flow sizes. More importantly, it allows study of the long-tail of long-lived flows, while reducing computation spent on the already-well-studied shorter flows (computation that would otherwise overwhelm analysis). In summary, sampling allows efficient observation of the correct trends, but not absolute density scales across durations.

4.1.1 Flow Rates

We first look at flow rate vs. duration in Figure 5. We see that short-duration flows can be quite fast, spanning 6 orders of magnitude speed. By contrast, long flows are typically much slower. Quartiles show median rates are around 50 bytes/s for flows shorter than 40 minutes, with a broad distribution, while flows longer than 100 minutes or longer have medians closer to 10 bytes/s.

The slower rate of long flows may be helpful for traffic engineering, allowing longer time to react to long-lived but slow-moving flows. Although we see very different rates at different durations, rate alone does not show which flows contribute to traffic. To evaluate if “slow and steady wins the race”, we next look at flow sizes across all time.

4.1.2 Flow Sizes

Prior studies of “slow but steady” tortoise flows can account for significant traffic [5, 6]. Having just shown long flows are slower than short flows, we next consider if their persistence makes up the difference.

Figure 6 shows the flow sizes (in bytes) of D8. We see a strong correlation between flow duration and total number of bytes at a slower-than linear rate on the log-log plot. Linear regression of median shows an exponentially increase at a rate of 0.77 with a 0.958 confidence coefficient.

Although each long-duration flow sends many bytes, there are many more brief flows, so in the aggregate short flows may still dominate traffic by bytes. Figure 7 shows the cumulative number of bytes sent by *all* flows of a two day period in D1. (Unlike density plots, this CDF considers all flow segments of all time scales sent over the entire period.) This graph confirms that there are not enough long-duration flows to dominate Internet traffic. From the figure we can observe that although the short flows dominate the Internet traffic (in terms of bytes), 21.4% of the Internet traffic are carried by flows longer than 10 minutes, 12.6% are carried by flows longer than 20 minutes, and nearly 2% are carried by flows longer than 100 minutes. Even though short flows are the majority of traffic bulk, optimizations to long flows can still have a significant effect. Internet Service Providers may also be interested in this observation, since the contribution of long-running but slow flows supports the need to meter service by bytes, not by peak speeds.

4.1.3 Flow Burstiness

Burstiness measures the uniformity of traffic rate over time. From Figure 9, we can observe that long flows are generally less bursty than short flows (linear regression of me-

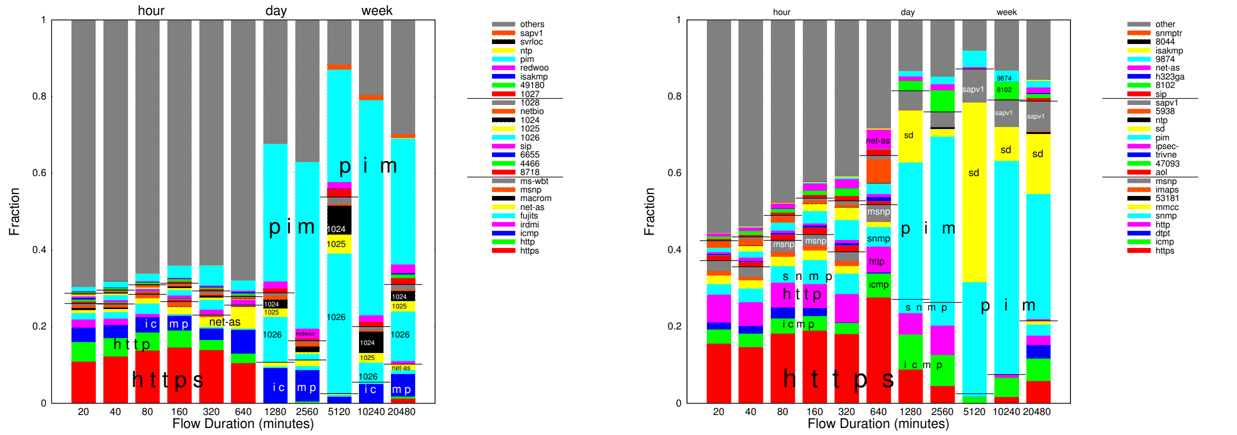

Figure 8: Source (left) and destination (right) port usage, plus PIM and ICMP, as a function of time scale (sampled from D8). Well-known ports are labeled with protocols and protocol colors differ.

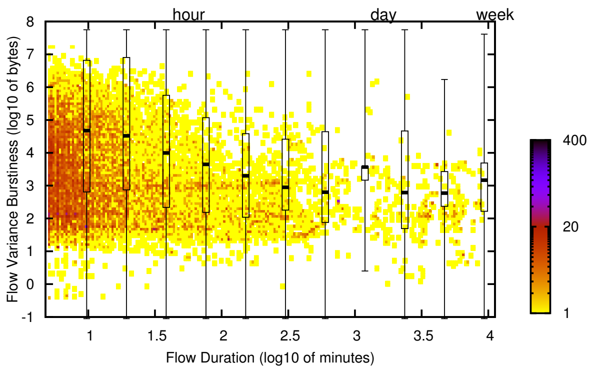

Figure 9: Density plot (log-scale) of flow duration vs. burstiness (log-scale variance, bytes) (sampled from D8).

dian shows an exponentially decrease at rate -0.296, with a -0.830 confidence coefficient). Our explanation, confirmed when we consider causes in Section 4.2, is that long flows are mostly computer-to-computer communications, and so are naturally less bursty. One implication of this observation is for very low-buffer routers, which assume input traffic is smooth [1,21]. Low burstiness could be provided with high probability in segregated long-duration flows.

4.2 Causes of Long-lived Flows

While we observe long-duration flows behave differently, we would like to know their causes. Although imperfect, port-based classification is our best tool and Figure 8 shows fraction of flows by port usage from minutes to weeks. We treat ICMP and Protocol Independent Multicast (PIM) as special "ports".

The result supports our hypothesis: the traffic mix switches from interactive, to background, to computer-to-computer as time scale increases. Hour time-scales are dominated by web (HTTP, HTTPS, ports 80 and 443) destinations. Web is also a frequent source port as well. Although at first it may seem suprising to think about port 80 as a source of traffic, this observation follows because our analysis treats each side of a bidirectional connection independently, so a port-80 source is the reply-side of a web request Day-long flows contain "background" traffic, with computer-driven but human-initiated protocols like chat and messaging (msnp, aol). We believe these represent regular application-level keep-alives and presence reports. Finally, week-long flows are almost all computer-to-computer protocols that run without human involvement, such as time synchronization (ntp) and multicast control (sd, pim, sapv1). This trend also shows with a very strong shift against TCP in the *protocol* mix at longer time scales: TCP is 66% through 10 hours, but falls to 16% at two weeks, where 30% is PIM and 43% UDP.

Another interesting result is that ports 1024 through 1026 are very common sources for long-lived flows. These are the first non-reserved ports and we believe indicate long-running, started-at-boot daemons.

Although we have identified the question of causes for long-lived Internet flows, we have only preliminary answers. Port-based classification schemes are well known to be inaccurate as many protocols today intentionally use random ports, so use of other techniques to identify applications is one direction (potentially those of Kim *et al.* [15]). Also, carrying out similar experiments in other locations, and more thorough evaluation of causes of long-running flows (protocols or applications) are both important future directions.

5. CONCLUSION

We propose an efficient multi-time-scale IP flow analysis methodology in this paper, targeting at the long-lived flows. The characteristics of different time scales of flows have been studied, with flow duration ranging from minutes to weeks. Our results show that long-lived flows are generally slow running and non-bursty ingredients of the Internet traffic, which is useful for traffic engineering purposes. We also study the causes of the long-lived flows, and found that unlike short flows with much human traffic, they are mostly computer-to-computer traffic for specific application purposes.

Acknowledgments

We would like to thank Yuri Pradkin for developing and operating the LANDER infrastructure that enables this work. We are also grateful to the anonymous reviewers for many valuable comments on this paper.

6. REFERENCES

[1] Guido Appenzeller, Isaac Keslassy, and Nick McKeown. Sizing router buffers. In *Proceedings of the ACM SIGCOMM Conference*, pages 281–292, Portland, Oregon, USA, August 2004. ACM.

[2] Daniel Awduche and Yakov Rekhter. Multiprotocol Lambda Switching: Combining MPLS Traffic Engineering Control with Optical Crossconnects. *IEEE Communications Magazine*, 39(3):111–116, March 2001.

[3] Chadi Barakat, Patrick Thiran, Gianluca Iannaccone, Christophe Diot, and Philippe Owezarski. A flow-based model for internet backbone traffic. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop*, pages 35–47, Marseille, France, October 2002. ACM.

[4] Nevil Brownlee. Some Observations of Internet Stream lifetimes. In *Passive and Active Measurement Workshop*, pages 265–277, 2005.

[5] Nevil Brownlee and kc claffy. Understanding Internet traffic streams: Dragonflies and tortoises. *IEEE Communications Magazine*, 40:110–117, 2002.

[6] Kun chan Lan and John Heidemann. A measurement study of correlation of Internet flow characteristics. *Computer Networks*, 50(1):46–62, January 2006.

[7] Mark E. Crovella and Azer Bestavros. Self-similarity in world wide web traffic: evidence and possible causes. *ACM/IEEE Transactions on Networking*, 5(6):835–846, December 1997.

[8] Nick Duffield, Carsten Lund, and Mikkel Thorup. Estimating Flow Distributions from Sampled Flow Statistics. In *Proceedings of the ACM SIGCOMM Conference*, pages 325–337, Karlsruhe, Germany, August 2003. ACM.

[9] Cristian Estan, Ken Keys, David Moore, and George Varghese. Building a better NetFlow. In *Proceedings of the ACM SIGCOMM Conference*, pages 245–256, Portland, Oregon, USA, August 2004. ACM.

[10] Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, and John Zahorjan. Measurement, modelling, and analysis of a peer-to-peer file-sharing workload. In *Proceedings of the 19th Symposium on Operating Systems Principles*, pages 314–329, Bolton Landing, NY, USA, October 2003. ACM.

[11] K.-P. Ho, H. Dal, C. Lin, S.-K. Liaw, H. Gysel, and M. Ramachandran. Hybrid wavelength-division-multiplexing systems for high-capacity digital and analog video trunking applications. *IEEE Photonics Technology Letters*, 10:297–299, February 1998.

[12] Alefiya Hussain, Genevieve Bartlett, Yuri Pryadkin, John Heidemann, Christos Papadopoulos, and Joseph Bannister. Experiences with a continuous network tracing infrastructure. In *Proceedings of the ACM SIGCOMM MineNet Workshop*, pages 185–190, Philadelphia, PA, USA, August 2005. ACM.

[13] Qosient Inc. Audit Record Generation and Usage System (ARGUS). `http://www.qosient.com/argus/`.

[14] Sling Media Inc. Slingbox: a tv streaming device. `http://en.wikipedia.org/wiki/Slingbox`.

[15] Hyunchul Kim, KC Claffy, Marina Fomenkov, Dhiman Barman, Michalis Faloutsos, and KiYoung Lee. Internet traffic classification demystified: myths, caveats, and the best practices. In *CoNEXT '08: Proceedings of the 2008 ACM CoNEXT Conference*, pages 1–12, New York, NY, USA, 2008. ACM.

[16] H. T. Kung and S. Y. Wang. TCP Trunking: Design, Implementation and Performance. In *IEEE International Conference on Network Protocols*, page 222, Washington, DC, USA, 1999. IEEE Computer Society.

[17] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the self-similar nature of Ethernet traffic (extended version). *ACM/IEEE Transactions on Networking*, 2(1):1–15, February 1994.

[18] Sean McCreary and kc claffy. Trends in Wide Area IP Traffic Patterns - A View from Ames Internet Exchange. *CAIDA ITC Specialist Seminar*, 2000.

[19] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modelling TCP Throughput: A Simple Model and its Empirical Validation. In *Proceedings of the ACM SIGCOMM Conference*, pages 303–314, Vancouver, Canada, September 1998. ACM.

[20] Vern Paxson. Automated Packet Trace Analysis of TCP Implementations. In *Proceedings of the ACM SIGCOMM Conference*, pages 167–179, Cannes, France, September 1997. ACM.

[21] Anees Shaikh, Jennifer Rexford, and Kang G. Shin. Load-sensitive routing of long-lived IP flows. In *Proceedings of the ACM SIGCOMM Conference*, pages 215–226, Cambridge, MA, USA, September 1999. ACM.

[22] Kevin Thompson, Gregory J. Miller, and Rick Wilder. Wide-Area Internet Traffic Patterns and Characteristics (extended version). *IEEE Network Magazine*, 11(6):10–23, Nov/Dec 1997.

[23] USC. High-Performance Computing and Communications. `http://www.usc.edu/hpcc/`.

[24] Yin Zhang, Lee Breslau, Vern Paxson, and Scott Shenker. On the characteristics and origins of internet flow rates. In *Proceedings of the ACM SIGCOMM Conference*, pages 309–322, New York, NY, USA, 2002. ACM.