Modern Application Layer Transmission Patterns from a Transport Perspective*

Matt Sargent[1], Ethan Blanton, and Mark Allman[2]

[1] Case Western Reserve University, Cleveland, OH, United States
[2] International Computer Science Institute, Berkeley, CA, United States

Abstract— We aim to broadly study the ways that modern applications use the underlying protocols and networks. Such an understanding is necessary when designing and optimizing lower-layer protocols. Traditionally—as prior work shows—applications have been well represented as bulk transfers, often preceded by application-layer handshaking. Recent suggestions posit that application evolution has eclipsed this simple model, and a typical pattern is now a series of transactions over a single transport layer connection. In this initial study we examine application transmission patterns via packet traces from two networks to better understand the ways that modern applications use TCP.

1 Introduction

In this study we seek to broadly understand the ways that modern applications use the underlying protocols and networks. In particular, we are interested in the transmission patterns of applications as viewed at the transport layer. While previous studies have documented these issues to some degree, we are motivated by the following two points.

- We aim to ensure that our mental models of application-imposed behavior are up-to-date. For instance, [14] suggests that while application behavior varies, when simulating Internet traffic a reasonable rule of thumb is to use connection sizes described by the log-normal distribution. In other words, a TCP connection is established, a given number of bytes sent, and then the connection is torn down. This behavior approximates traditional applications like HTTP/1.0 and FTP. However, some in the community have stated their belief that applications' use of TCP has evolved to a more transaction-oriented nature wherein an application re-uses connections for a number of small transactions (e.g., as part of a web application) [5].
- Second, good network engineering crucially depends on an empirical understanding of the system. For instance, intrusion detection systems must understand the difference between an abandoned connection and a quiescent application. Another example is understanding the importance of the so-called "last window" problems in TCP (e.g., [6]). The amount of justifiable additional complexity in TCP to deal with such problems depends on whether there is one "last window" in a connection (e.g., the bulk transfer case) or there are numerous "last windows" (e.g., at the end of every transaction in a connection with many transactions).

As an initial check on these two points we examine packet traces from the Lawrence Berkeley National Laboratory (LBNL) and the International Computer Science Institute

* Work supported in part by NSF grants CNS-0831535 and CNS-1213157.

	CCZ	ICSI
Time	2/11–3/12	9/12–3/13
Length (hrs)	98	1,176
Total Conns.	6.5M	56.9M
Conns. w/o Data	2.6M	27.9M
Port Filtered	-	1.4M
Remaining	3.9M	27.6M

Table 1: Data overview.

Location	CCZ	ICSI
No N	31%	51.2%
Internal-only	14.4%	18.3%
Trailing-only	32.3%	20.7%
Internal & Trailing	22.3%	9.8%

Table 2: Prevalence of N periods at various positions.

(ICSI). For each connection we compute the maximum duration between data segments. Bulk transfers would tend to show sub-second gaps, while multiple distinct transactions would likely show a larger maximum gap driven by application behavior. We find that in both datasets, the *proportion* of connections with maximum gaps of more than one second and the *duration* of the gaps increases over time. In the LBNL dataset roughly 55% of the connections have a maximum silent period of at most 275 msec in both 2003 and 2013. The distributions then diverge with 4% more connections containing a gap of at least 1 second in 2013 than in 2003 and 12% more connections having a gap of at least 10 seconds. Similarly, in the ICSI data, the distribution of the maximum gap per connection is similar for 2007 and 2013 data up to 1 second—covering about two-thirds of the connections. However, 13% more connections have a maximum gap of at least 10 seconds in 2013 than in 2007. While this analysis is simple and anecdotal it suggests an in-depth exploration of modern application behavior is warranted.

We use packet-level traces from two vantage points—a small research laboratory and a small residential network—as the basis of an initial study into application patterns from TCP's perspective. We contribute both an application agnostic methodology and an initial understanding of modern TCP-based applications.

2 Related Work

There are two general classes of related work. First, there is a vast and long-standing vein of work that characterizes and models specific application protocols. These studies span much time and many protocols, from the largely outdated (e.g., [12]) to a rich understanding of early web traffic (e.g., [3, 4]) to modern applications (e.g., [17]). A second class of previous work attempts to identify applications based on the behavior they exhibit on the network (e.g., [8, 9]). We do neither of these things, preferring to understand the traffic patterns applications impose on the transport protocol.

3 Data

We analyze the two sets of packet traces summarized in Table 1.[3] The first dataset is gathered from the border of a residential fiber-to-the-home network, the Case Connec-

[3] Note, the LBNL data we present in § 1 is anecdotal in that each trace covers only a single hour. We believe it is useful for motivating the problem, it is not sufficient for deeper analysis and therefore not used in the remainder of the paper.

tion Zone (CCZ) [1]. The CCZ connects roughly 90 residences (200-300 users) with bi-directional 1 Gbps fiber. While the connection is abnormal for US residential users, we find in previous work that actual use of the bandwidth is modest—topping out at roughly 10 Mbps in the typical case—and the application mix is in line with previous studies of residential network users [15]. Our second dataset is gathered from the border of the International Computer Science Institute, and covers roughly 100 users. In both cases we gather data between the 11^{th}–17^{th} of each month. We capture all packets from our ICSI vantage point. Our measurement capabilities within the CCZ network are more modest and we collect a one-hour trace from a random time for each day. As we develop in more detail in [15], the CCZ measurement apparatus does not often drop packets during the collection process, with no detectable measurement-based loss in the majority of the traces and the loss rate reaching 0.013% in the worst case. The tracing apparatus at ICSI experiences more measurement-based loss than the CCZ monitor, with an average loss rate of roughly 2.1%. We account for measurement-based loss in our analysis by either not considering missing packets or inferring their existence (by noting progression of TCP's sequence space for missing packets), as appropriate.

We prune the datasets before use for two reasons. First, we do not consider connections that do not have at least one byte of data flowing form the monitored network to the remote network. This rule largely removes scanning and backscatter. Further, in the ICSI dataset we noticed two large traffic anomalies that turned out to be part of an independent experiment: (i) a large crawl of the *whois* databases and (ii) a large backhauling of data to Amazon's EC2. These activities are sufficiently voluminous to affect our results. Therefore, since this traffic is also abnormal, we filter it from further analysis. Table 1 shows the number of connections we remove from further analysis.

4 Dividing Connections

Our general strategy for analyzing application behavior is to take stock of the amount and temporal location of silence in TCP connections. Under this model, traditional bulk data transmission would show few instances where a connection was not actively transmitting data in one or both directions except at the beginning and end of a connection. Of course, our approach is not fool-proof. For instance, streaming may look like bulk transfer in that there are few silent periods, but may be pushing only as fast as required for the given media and not as hard as a bulk transfer. While this is also an important aspect of application behavior to understand, we leave it for future work.

Given our data, we do not have details of the precise application operations. Additionally, our lack of application payload precludes a study based on application protocol semantics.[4] We approximate application behavior with the following process:

ON/OFF Periods: As a first cut we divide connections into ON and OFF periods with respect to the transmission behavior of the local host (the host close to our monitor) in the connection. Each connection begins in an OFF period and transitions to an ON period when we observe the local host sending a data segment. Transitioning from an ON period to an OFF period happens when two conditions are met: (i) all outstanding data

[4] Additionally, encrypted traffic is not amenable to such analysis.

sent by the local host is acknowledged (ACKed)[5], and (ii) either the local host sends an ACK containing no data or at least 5 msec passes without the local host sending another data segment. Note that once we are in an ON period we are able to deal with loss from the local host by advancing the TCP sequence number based on local packets being sent after the loss or by noticing a gap in the sequence space once rule (i) is met and all of the outstanding data has been ACKed. Lost packets during an ON period will not change the length of the ON period that we detect. Rule (ii) ensures that the local TCP does not have application data waiting to be sent. A bare ACK indicates directly that the TCP buffer is empty. The 5 msec rule is otherwise necessary to account for TCP's slow start behavior [7, 2]. Consider a local host that sends a single segment; when that segment is ACKed, criteria (i) is met. However, in slow start, we expect the local host to use the ACK to open the congestion window and transmit additional data. Therefore, data coming within a short amount of time should be considered part of TCP's dynamics and not part of the application's dynamics. We studied the length of the OFF periods without criteria (ii) to find a reasonable threshold, and thresholds of 1–10 msec show similar results. The 5 msec threshold is a somewhat arbitrary choice within that range.

Refinement: Two-Way Traffic: The ON/OFF analysis only accounts for traffic in one direction (from the local to the remote). This approach does not reveal the applications' full complexities, but reconstructing the TCP state of hosts distant from a monitor is known to be difficult [13]. Therefore, we use the following heuristics to glean enough information about returning data to conduct our analysis without reconstructing the entire state of the remote host. We couple the ON/OFF classification above with information about the data flow from the remote host to the local host to refine our classification into four types: Local-only periods are ON periods where we do not observe data sent by the remote host, Remote-only periods are OFF periods where we observe data sent by the remote host, Both periods are ON periods where we also find data sent by the remote host, and None periods are OFF periods where we find no data sent from the remote host. N periods are a first approximation of the silent periods we describe at the beginning of this section. We find that R periods hide silence at times. Consider the case where a single data segment is sent from the remote just after the start of an OFF period and then the connection goes silent for a long period of time. In this case, we classify the entire period as R, when most of the period is in fact silent. We remedy this by terminating an R period—at the point of the last data segment arrival—if twice the minimum observed RTT for the connection elapses without another data segment from the remote host. Twice the minimum RTT provides some robustness to network and TCP behaviors while ensuring that the model transitions in a timely fashion. An N period is inserted for the remaining duration of the shortened R period. R periods that do not trigger this rule may still contain some silence, but the duration of this error is bounded by twice the minimum RTT. Together, these heuristics provide a conservative estimate of the silent periods. Any N period in the analysis is a true silent period, but there may be short application silences hidden in L, R, or B periods.

As a next step, we build a map for each connection that consists of a string corresponding to the order of the various periods in the connection. For instance, a map of

[5] Note, this criteria naturally keep original transmissions and their retransmissions in the same period.

NLR indicates an initial OFF period, then a period of local data transmission and the connection ending with a period of data transmission from only the remote host. We find over 155 k and 579 k unique maps within our CCZ and ICSI datasets, respectively. This shows the that applications display significant variety in their behavior. Over millions of connections, we find an average of 25 and 50 connections share each map in the CCZ and ICSI datasets, respectively. Further, we find that there are 12 "popular" maps, or maps that make up at least 1% of the connections, in the CCZ dataset and 10 popular maps in the ICSI dataset. Three maps—NBN, NLR and $NLRN$—are popular in both datasets. Popular maps account for a total of 63% of the connections in both datasets. These results underscore the vast heterogeneity in application behavior observed.

Next, we analyze where N periods fall within connections. Since many connections start with an N period following the three-way handshake due to TCP dynamics, we ignore initial N periods for this analysis. Table 2 shows the prevalence of N periods in various locations within the connection. First, we find that about one half to two thirds of the connections in both datasets contain periods where the application is silent. We believe this illustrates that the majority of the connections are not simple bulk transfers. Further, we find that of the connections with silent periods a plurality have only "trailing" silent periods (e.g., persistent HTTP keeping a connection open in case further requests are forthcoming, but ultimately closing with no such requests). Finally, we find that between a quarter and a third of the connections have an internal silent period, indicating an application pause. We present in-depth analysis in the next two sections.

5 Trailing Silent Periods

We first study trailing silent periods, or connections that transfer data and then go silent before terminating. Persistent HTTP follows this model, as connections speculatively persist after the "final" response in case the browser subsequently needs more objects. This mechanism aids performance by allowing subsequent transactions to avoid the overhead of starting a new connection [11]. As we note above, 54.6% and 30.5% of connections from CCZ and ICSI, respectively, end with a silent period. Note that these connections may not violate the bulk transfer model of TCP behavior, as they may behave as bulk transfers that simply do not close immediately when activity completes.

The left plot in Figure 1 shows the distribution of the duration of trailing silent periods. Trailing silence of less than 1 second happens in about 30% and 20% of the connections for CCZ and ICSI, respectively. These likely represent applications finishing processing tasks before closing the connection. On the other hand, we find that just under half of the trailing silent periods last longer than 10 seconds in both datasets. This likely indicates the application speculatively leaving a connection open in case further work materializes—which never happens in these cases. These trailing silent periods can be lengthy, with nearly 20–25% of the periods extending beyond 2 minutes. Further, 10% of the trailing silent periods exceed 4 minutes in each dataset.

We next study the behavior of specific applications [6] with respect to trailing silent periods. The right plot in Figure 1 shows the characteristics of each port that contributes

[6] Our traces include only packet headers and therefore we rely on port numbers to identify applications—as crude as that can sometimes be.

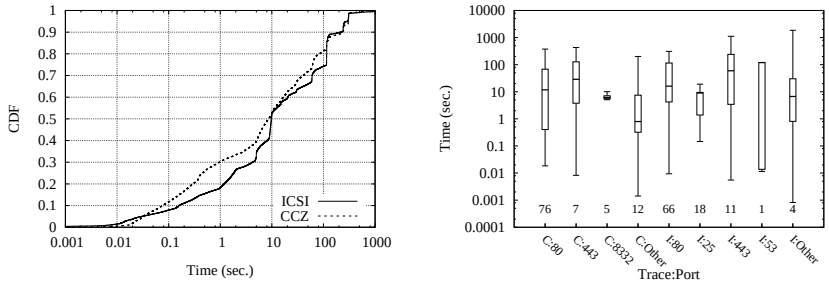

Fig. 1: Duration of trailing N periods.

at least 1% of the connections with trailing silent periods. The labels on the x-axis indicate the dataset—"C" for CCZ, "I" for ICSI—and port number for the applications, with "other" being a combination of all ports not shown independently. The number just above the x-axis shows the percentage of connections with trailing silent periods that the given port is responsible for in the given dataset. For each port, the box shows the quartiles of the distribution of the duration of the trailing silent periods and the whiskers show the 1st and 99th percentiles.

The figure shows that at least three-quarters of the connections with trailing silent periods across datasets are likely web traffic (ports 80 and 443) and web traffic generally shows the longest trailing silent periods. Additionally, we find three times as much "other" traffic in the CCZ data as in the ICSI data. This is natural in that CCZ traffic contains more peer-to-peer traffic that is widely distributed across the port range and therefore confounds such simple port-based classification (see [15] for details). We find that CCZ traffic using port 8332[7] has short and highly uniform trailing silent periods. The "other" traffic generally has the largest spread of trailing silent periods, as one might expect, given that it is an amalgamation of different applications. The ICSI dataset includes many SMTP connections with trailing silent periods; while half of these are at least 10 seconds, the 99th percentile is only 19 seconds, which suggests that a fairly tight timeout is in play. Finally, we find that TCP-based DNS traffic in the ICSI dataset is responsible for roughly 1% of the trailing silent periods. Two ICSI hosts are responsible for most of this DNS traffic, and the general pattern of their connections is consistent with a single, short DNS lookup followed by a 2 minute timeout—which is consistent with the behavior specified in RFC 1035 [10].

6 Internal Silent Periods

Our next analysis is of silent periods that happen between periods of activity within connections. These periods indicate an application imposing a non-bulk transfer structure on their activity. There could still be periods in which the application—and therefore

[7] As discussed in [15], we have not been able to fully disambiguate this traffic between Bitcoin and an experimental security camera application known to be in use within the CCZ.

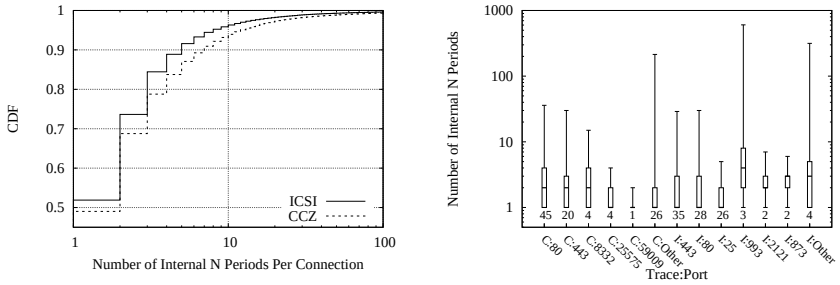

Fig. 2: Number of internal N periods per connection.

TCP—tries to move data as fast as possible in bulk transfer fashion, but these silent periods indicate that is not the applications' exclusive goal.

Silent Periods Per Connection: Recall from Table 2 that 36.7% and 28.1% of the connections in the CCZ and ICSI datasets, respectively, contain at least one internal silent period. From this we understand that a non-trivial fraction of the connections are not solely concerned with bulk transfer. The left plot in Figure 2 shows the distribution of the number of internal silent periods per connection in our two datasets. We find general agreement between the datasets with roughly half the connections having only one internal silent period, and over 90% of the connections having no more than ten internal silent periods. Therefore, while we find that internal silent periods are not rare, we also find that they are in general not numerous on a per-connection basis.

The right plot in Figure 2 breaks down the number of silent periods per connection by port for ports that contribute at least 1% of the connections with internal silent periods. Again, the overall fraction of connections is given just above the x-axis, the bars represent quartiles and the whiskers show the 1st and 99th percentiles. We find that over 60% of the connections with internal silent periods in both datasets are web traffic (ports 80 and 443). Further, most of the popular ports have a median of one internal silent period per connection and the 75th percentile is under 10 periods across ports. This is consistent with the overall distribution given in the left figure and shows that popular ports do not drastically depart from the overall distribution. We do find that IMAP connections at ICSI (port 993) show a large 99th percentile—604 silent periods. This is expected for email clients that leave connections open for pushed email.

Silent Period Duration: We next assess the duration of internal silent periods, as we show in Figure 3. This plot shows that most such periods are short—with at least 30% lasting at most 100 msec and two thirds lasting at most 1 second. These durations are consistent with the "active off" periods previously identified in web traffic [4]. However, more than 10% of the internal silent periods across connections last at least 10 seconds. These periods likely represent applications that run out of networking tasks.

The duration of internal silent periods is not as uniform across applications as their number, as shown in Figure 3. For example, SMTP (port 25) is largely rapid exchanges, with 75% of silent periods lasting less than about 100 msec and no silent period lasting more than a few seconds. On the other hand, web traffic (ports 80 and 443) show sig-

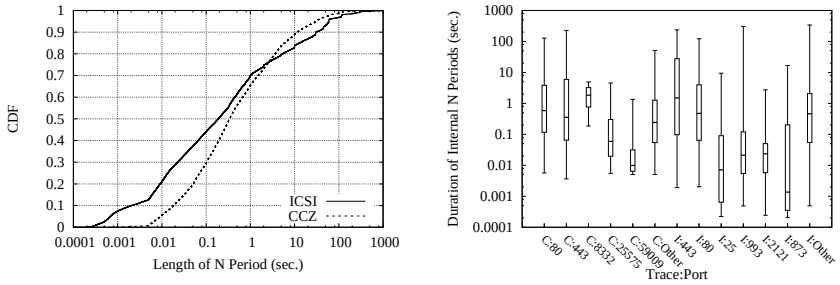

Fig. 3: Duration of internal N periods.

nificantly longer internal silent periods in both the ICSI and CCZ traces. Interestingly, we note that port 443 has longer internal silent periods than port 80 in both datasets—but more exaggerated in the ICSI dataset. We speculate that this may be due to more aggressive caching of HTTPS connections to avoid the higher setup cost of SSL/TLS.

We now turn from focusing on individual internal silent periods to the amount of aggregate silence we find across an entire connection. We calculate the total fraction of each connection with least one internal silent period that is spent in silence. We find that two thirds of the connections are fairly uniformly distributed between nearly no silence and roughly 90% silence across the connection. However, in the other one-third of the connections across datasets over 90% of the connection is silent—with roughly 20% of the connections in both datasets showing near total silence. The distribution of the number of silent periods for connections that are at least 90% silent shows that these connections have more silent periods than the overall distribution (which is shown in Figure 2)—indicating that a single silent period is not driving the overall behavior.

The Last Window Problem: TCP's loss recovery depends on the acknowledgment of packets received. The information in returning ACKs is used to drive retransmission decisions, by assuming that multiple incoming ACKs that do not acknowledge outstanding data indicate that the data was lost. However, ACKs are sent only when data is received, and there is no data after the last window to generate new ACKs. Hence, it is comparatively more difficult for TCP to determine that the final packets of a window have been lost; in many algorithms, this situation is detected only by a relatively long retransmission timeout (RTO). TCP also uses ACKs to trigger the transmission of new data. However, after a period of silence there are no incoming ACKs, and thus this "ACK clock" cannot be used to immediately pace out new data. This can lead to either a large burst of segments [7, 16] or the need to wait a full RTT for ACKs for the new data to return [16]. In other words, events that happen in a routine and timely fashion most of the time can be problematic at the "end" of a connection. A silent period within a connection can manifest the same behaviors.

Various proposals exist to deal with TCP's "last window" (e.g., [6]). However, understanding the frequency of this phenomenon is crucial to determining how much complexity should be added to TCP to deal with the issue. Our approach to assess this is to treat the window before a silent period as a "last window" as long as the silent pe-

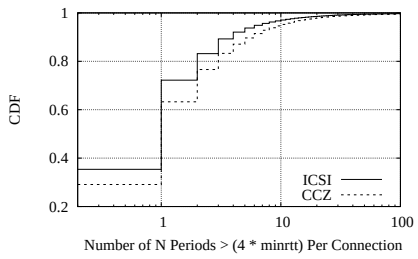

Fig. 4: # of N periods $>$ RTO.

Class	Med.	Mean	StdDev	# Cnns
CCZ Active	2	2.80	1.13	139k
CCZ Simple	3	3.45	1.34	2.5M
CCZ Complex	8	20.0	199	1.4M
ICSI Active	2	2.66	5.15	4.3M
ICSI Simple	4	4.79	3.88	19.8M
ICSI Complex	8	27.2	714	7.8M

Table 3: Length and diversity of connection maps.

riod is relatively long, which we define as roughly the length of an RTO. We use this approximation because of the recommendation that TCP collapse its congestion window after an RTO worth of idle time [2]. Since the specifics of the RTO vary across implementations we use $4 \times minRTT$ as an approximation.

We find that 65–71% of the connections have internal silent periods that last at least $4 \times minRTT$—which represents at least a doubling of last windows (i.e., one internal and one actual last window). Figure 4 shows the distribution of the number of silent periods that exceed $4 \times minRTT$ per connection. We find that 32% and 24% of the connections that have internal silent periods for CCZ and ICSI, respectively, have 2–10 silent periods of at least $4 \times minRTT$. These results show that a non-trivial number of connections would benefit from techniques that mitigate last window issues.

7 Application Complexity

We next assess the diversity of patterns of activity within connections. For this analysis, we classify connections into three types: (i) "active" connections consist only of L, R, and B periods, with no N period, (ii) "simple" connections may have initial and/or trailing N periods, but all other periods must be L, R, or B (note that active connections are a subset of simple connections) and (iii) "complex" connections which may have any combination of periods. Table 3 shows a summary of our analysis. The data suggests that active and simple connections are much more likely to consist of a small number of exchanges followed by termination, whereas complex connections—those with at least one internal N period—display a large diversity of internal structure, involving a comparatively larger number of exchanges and period transitions.

The tendency of simple connections to be classic bulk transfers is strong. Out of the CCZ simple connections, 90% of the maps (2.2M connections) consist of no more than two periods containing data—with 60% being LR, with or without initial and trailing N periods—suggesting a simple request-response bulk transfer. The ICSI data is somewhat more diverse, with the corresponding maps accounting for 47% of the simple connections. Further, 40% of the connections are either LR or RL with or without initial and trailing N periods. This suggests that the simple connections in the ICSI dataset are

somewhat more complicated than in the CCZ dataset, but the overall diversity remains markedly lower than for complex connections.

8 Conclusions

This paper makes several initial contributions: (i) we provide an application agnostic methodology for studying application patterns from the transport's perspective, (ii) we confirm that TCP is non-trivially used for non-bulk transfer applications, which breaks our often-employed mental model, (iii) while silent periods within connections exist, they are mostly short, (iv) we find that TCP's "last window" problem is exacerbated by the transactional nature of some connections and (v) we find that connections with internal silent periods have more complicated interactions than those without such periods. We stress that this is an initial investigation and the results in some sense offer more questions than answers—which we are grappling with as future work.

References

1. Case Connection Zone. `http://caseconnectionzone.org/`.
2. M. Allman, V. Paxson, and E. Blanton. TCP Congestion Control, Sept. 2009. RFC 5681.
3. M. Arlitt and C. Williamson. Web Server Workload Characterization: The Search for Invariants (Extended Version). *IEEE/ACM Transactions on Networking*, 5(5), Oct. 1997.
4. P. Barford and M. Crovella. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *ACM SIGMETRICS*, July 1998.
5. Y. Cheng. Re: [tcpm] Adopting draft-fairhurst-tcpm-newcwv. IETF TCPM Mailing List, Dec. 2012.
6. N. Dukkipati, N. Cardwell, Y. Cheng, and M. Mathis. TCP Loss Probe (TLP): An Algorithm for Fast Recovery of Tail Losses. Internet-Draft draft-dukkipati-tcpm-tcp-loss-probe-00.txt, July 2012. Work in progress.
7. V. Jacobson. Congestion Avoidance and Control. In *ACM SIGCOMM*, 1988.
8. T. Karagiannis, K. Papagiannaki, and M. Faloutsos. BLINC: Multilevel Traffic Classification in the Dark. In *ACM SIGCOMM*, 2005.
9. H. Kim, k. claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee. Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices. In *ACM SIGCOMM CoNEXT*, Dec. 2008.
10. P. Mockapetris. Domain Names - Implementation and Specification, Nov. 1987. RFC 1035.
11. H. Nielsen, J. Gettys, A. Baird-Smith, E. Prud'hommeaux, H. Lie, and C. Lilley. Network Performance Effects of HTTP/1.1, CSS1, and PNG. In *ACM SIGCOMM*, Sept. 1997.
12. V. Paxson. Empirically-Derived Analytic Models of Wide-Area TCP Connections. *IEEE/ACM Transactions on Networking*, 2(4), Aug. 1994.
13. V. Paxson. Automated Packet Trace Analysis of TCP Implementations. In *ACM SIGCOMM*, Sept. 1997.
14. V. Paxson and S. Floyd. Difficulties in Simulating the Internet. *IEEE/ACM Transactions on Networking*, 9(4):392–403, 2001.
15. M. Sargent, B. Stack, T. Dooner, and M. Allman. A First Look at 1 Gbps Fiber-To-The-Home Traffic. Technical Report 12-009, International Computer Science Institute, Aug. 2012.
16. V. Visweswaraiah and J. Heidemann. Improving restart of idle TCP connections. Technical Report 97-661, University of Southern California, November 1997.
17. Y. Xu, C. Yu, J. Li, and Y. Liu. Video Telephony for End-consumers: Measurement Study of Google+, iChat, and Skype. In *ACM Internet Measurement Conference*, Oct. 2012.