Efficient Policies for Carrying Web Traffic Over Flow-Switched Networks

Anja Feldmann, Jennifer Rexford, and Ramón Cáceres

***Abstract*— To efficiently transfer diverse traffic over high-speed links, modern integrated networks require more efficient packet-switching techniques that can capitalize on recent advances in switch hardware. Several promising approaches attempt to improve performance by creating dedicated "shortcut" connections for long-lived traffic flows, at the expense of the network overhead for establishing and maintaining these shortcuts. The network can balance these cost-performance tradeoffs through three tunable parameters: the granularity of flow end-point addresses, the timeout for grouping related packets into flows, and the trigger for migrating a long-lived flow to a shortcut connection. Drawing on a continuous one-week trace of Internet traffic, we evaluate the processor and switch overheads for transferring HTTP server traffic through a flow-switched network. In contrast to previous work, we focus on the full probability distributions of flow sizes and cost-performance metrics to highlight the subtle influence of the HTTP protocol and user behavior on the performance of flow switching. We find that moderate levels of aggregation and triggering yield significant reductions in overhead with a negligible reduction in performance. The traffic characterization results further suggest schemes for limiting shortcut overhead by temporarily delaying the creation of shortcuts during peak load, and by aggregating related packets that share a portion of their routes through the network.**

Keywords: **Internet traffic characterization, IP flows, HTTP protocol, switching, signaling, routing.**

1 Introduction

The explosive growth of Internet traffic adds urgency to the search for more efficient packet-switching techniques. To exploit recent advances in high-speed switch hardware, several proposals call for grouping sequences of related IP packets into *flows*, and sending them through fast switching paths, or *shortcuts*, through the network fabric [1, 4, 7, 19, 25, 27, 31, 33]. Establishing dedicated connections for long-lived flows improves performance by reducing packet-forwarding load on the routers and capitalizing on quality-of-service features in the underlying switching hardware, and can help balance network load by directing shortcuts to underutilized links. Despite these potential benefits, shortcutting also consumes network resources to create and maintain a dedicated connection. In this paper we explore the tradeoffs inherent in using different definitions for what constitutes a flow and different criteria for creating and maintaining a shortcut connection. We look at ways to reduce overhead in the network while still giving a majority of traffic the benefits of following a shortcut. The remaining packets are routed and switched along a default path.

We consider three parameters that determine flow and shortcut decisions: *aggregation*, *timeout*, and *trigger*. Aggregation refers to the addressing level at which traffic is combined to form a flow. For example, a flow may be defined to contain all traffic flowing between two IP hosts (i.e., host-to-host flows). Alternatively, a flow may contain only traffic flowing between two application processes running in two IP hosts (i.e., port-to-port flows). Timeout refers to how long a flow is idle before it is considered closed. For example, a flow may be defined as closed if no traffic at the chosen aggregation level has been detected in the previous 60 seconds. Trigger refers to how much traffic appears in a flow before a shortcut is established for that flow. For example, a shortcut may be established only after 10 packets that obey a flow definition have been detected. The selection of these three parameters directly affects the ability of the network to detect and shortcut long-lived traffic.

We explore the effects of varying these parameters on three metrics of interest: *the percentage of traffic that follows shortcuts*, *the shortcut setup rate*, and *the number of simultaneous shortcuts*. The percentage of total traffic that follows shortcuts is a measure of the performance gains achieved by using shortcuts. The higher this metric, the higher the performance. In contrast, the shortcut setup rate and the number of simultaneous shortcuts are measures of network overhead resulting from using shortcuts. The higher these two metrics, the higher the overhead. In contrast to previous studies, we characterize the *distribution* of these metrics to study the network load on a variety of time scales, in order to develop an understanding of how the traffic dynamics affect network overhead, in order to develop effective guidelines for the design and provisioning of the network.

We base our study on temporal and spatial characteristics of Internet traffic. There is a growing body of work in measurement-based traffic characterization and its application to network design [2, 8, 11, 16, 18, 20–22, 25, 30, 32]. Our effort extends previous work in several ways. First, our traffic traces reflect the dominance of World Wide Web traffic in today's Internet. We focus on Web traffic and take a network-centric view, specifically not focusing on either client or server traffic. Second, we use long, continuous traffic traces. Working with a full week of data allows us to study the effects of looking at traffic on different time scales such as minutes, hours, and days. Third, we differ-

A. Feldmann, J. Rexford, and R. Cáceres are with the Networking and Distributed Systems Research center at AT&T Labs – Research in Florham Park, New Jersey.

entiate between endpoint types. We identify in our traces which Web requests are coming from proxy servers, multi-user clients, and single-user clients on modems and local area networks. Finally, we consider the interaction of all three flow and shortcut parameters on a variety of time scales.

The following summarizes our main observations and their implications:

- *Probability distributions of Web flow sizes have several modes:* The distribution of the number of bytes in Web response flows depends heavily on the HTTP protocol and the distribution of Web document sizes. The flow-size distributions have three modes corresponding to error conditions (10% of responses), cache validation messages (18%), and actual data transfers (72%). The network can use a large enough trigger to avoid establishing separate shortcuts for the error and cache validation messages, while still carrying most of the data transfers on shortcut connections.
- *Flow characteristics vary with endpoint type:* The flow-size distributions depend on whether the client endpoints are modems, personal computers, compute servers, or proxy servers. For example, flows to modem-connected clients typically have fewer bytes but longer durations. On average, flows to shared UNIX machines have more bytes and packets, mainly due to a small number of very large flows. Therefore, a network may want to have different shortcut policies for different endpoint types.
- *Aggregating consecutive and concurrent transfers from the same Web server yields substantial benefits:* Combining multiple transfers between a Web server and a client reduces the number of flows on average by a factor of five. Host-to-host aggregation captures the transfer of embedded images and, in some cases, the browsing of multiple pages at the same site. In addition to reducing shortcut overheads, aggregation also reduces the unfairness that can result when a single client establishes multiple TCP sessions to the same server. These effects also preview some of the benefits of the persistent connections in HTTP 1.1.
- *Aggregation and triggers both reduce overhead, but the reductions are not multiplicative:* Aggregation results in longer-lived flows, which negates some of the impact of triggers. For example, applying a 10-packet trigger to port-to-port flows reduces the shortcut setup rate by a factor of 3.3. Aggregating port-to-port flows to the host level reduced the setup rate by a factor of 5.2. Yet, the combination of host-level aggregation and a 10-packet trigger offers only a factor 8.9 reduction over port-to-port flows with a 0-packet trigger, notably less than a multiplicative gain.
- *Variability in traffic load changes with time scale:* For example, our traces show that there are large variations in the peak load between the 10-second and 100-second time frames, but there are many periods of consistent maximum load when viewed from a 1- to 2-hour time frame. Our trace shows that the peak shortcut setup rate differs by a factor of two between the 10-second and 100-second timescales. Based on these results, we propose techniques that adjust to short-term variations by temporarily delaying the establishment of shortcut connections, while provisioning network signaling resources based on the longer time scale requirements.
- *Aggregating traffic along portions of the route between the source and destination yields additional reductions in network overhead:* By combining traffic from server replicas and other nearby sites into a single flow, the network can reduce shortcut overheads without having to aggregate traffic from unrelated users. In addition, partial-route aggregation allows more of the packets to travel on a shortcut connection. For example, by aggregating host-to-host flows that have the same first seven hops along their paths from the clients, we see a reduction of 20% in the setup rate and 11% in the number of simultaneous connections, while increasing the number of packets on shortcut connections by 2%.

The remainder of the paper is structured as follows. Section 2 describes our traffic measurement environment and an initial characterization of the packet-level trace data. In Section 3, we focus on the probability distribution of flow sizes, in terms of the number of bytes and packets in a flow, as well as the time duration. These results highlight the unique flow dynamics in Web response traffic and the benefits of aggregating traffic at the host level. While the discussion in Section 3 is relevant to a variety of flow-switching schemes, Section 4 focuses specifically on the network overhead for creating end-to-end shortcuts for long-lived flows. The performance metrics highlight the network overhead on a variety of time scales to guide policies for provisioning network processing and switching resources. Then, Section 5 generalizes the flow definition to consider a *subset* of the path between the source and destination hosts. This allows the shortcutting mechanism to group packets that have a common route through a portion of the network, to further reduce the setup rate and number of simultaneous connections, while increasing the proportion of packets that travel on a shortcut connection. Finally, Section 6 concludes the paper with a discussion of future research directions.

This paper complements previous research on Web traffic, flow switching, and shortcutting architectures. Recent studies of Web traffic have typically focused on evaluating the HTTP protocol and its interactions with TCP, or comparing various caching and prefetching policies [3, 5, 6, 12, 24, 26]. Instead, we focus on understanding how HTTP and user behavior affect the performance trade-offs of shortcutting Web flows. Our detailed treatment of how the Web impacts the effectiveness of flow switching contrasts this paper with other recent work on evaluating IP flow characteristics and shortcutting policies [10, 11, 22, 25, 32]. In addition, we consider the full distribution of the shortcut metrics applied to a long, continuous trace. Finally, our study relates to the numerous proposals for carrying IP traffic on ATM networks [1, 4, 7, 19, 25, 27, 31, 33] by evaluating the effectiveness of these schemes, and suggesting two new techniques to reduce the overhead of establishing shortcuts. Section 4.2 describes how to limit signaling overheads by delaying the establishment of shortcut connections is appropriate for any flow-switching scheme that provides a default forwarding path for packets. The partial-route aggregation techniques in Section 5.2 are appropriate for flow-switching schemes that allow a single shortcut connection to cross a collection of switches.

2 Packet Trace Collection

The IP flow characterization draws on a continuous, one-week packet-level trace of Internet traffic. The trace contains diverse traffic to and from a mixture of end-point machines, including office personal computers, shared UNIX machines, proxy servers, and modem-connected hosts.

2.1 Trace Collection

For a realistic study of IP flow characteristics, we have collected extensive packet-level traces of the traffic on the T1 line that connects the AT&T Research to the external Internet. A single 10 megabit/second Ethernet segment carries all traffic to and from this T1 line, as shown in Figure 1, permitting efficient trace collection using `tcpdump` [17] on a personal computer operating in promiscuous mode. To collect long traces, each tcpdump output file contains header and timestamp information for 100,000 packets in a raw binary format. In the background, a Perl script compresses these output files and moves them to a multiprocessor compute server for more extensive post-processing. Using tcpdump, this machine reads each binary file to produce a condensed ASCII log that records the relevant information for each packet. Each entry includes a receive timestamp, along with the source and destination IP addresses and port numbers in the packet header.

Based on this information, an additional Perl script classifies packets into flows based on a timeout value, as well as a mask on the end-point IP addresses and port numbers. These masks can aggregate sessions with the same port, host, subnet, or net addresses. After applying the source and destination masks, the script can group packets with matching end-point addresses into a single flow, with the timeout value determining the maximum spacing between consecutive packets in the same flow. The script records starting time (the receive timestamp of the first packet), duration (based on the receive timestamps of the first and last packets), total number of packets, and total number of bytes (including the IP headers) for each flow. Then, various Splus functions process these log files to compute performance metrics at the flow level.

Most of the experiments in this paper evaluate a subset of the trace data to measure the flow statistics for specific types of traffic. For example, although the Ethernet segment mainly shuttles traffic to and from the external Internet, a small portion of the packets have both their source and destination IP addresses on the local AT&T Research subnet. This internal traffic accounts for approximately 7.5% of the bytes and 21% of the packets, mostly from short Domain Name Service (DNS) messages. To focus our study on Internet flow characteristics, we have removed the internal traffic from the trace and have classified the remaining packets as incoming or outgoing packets, based on whether the source or the destination is an external IP address. The majority of the paper focuses on the incoming Internet traffic, which represents 76.1% of the bytes on the Ethernet segment, compared to 16.4% for outgoing traffic.

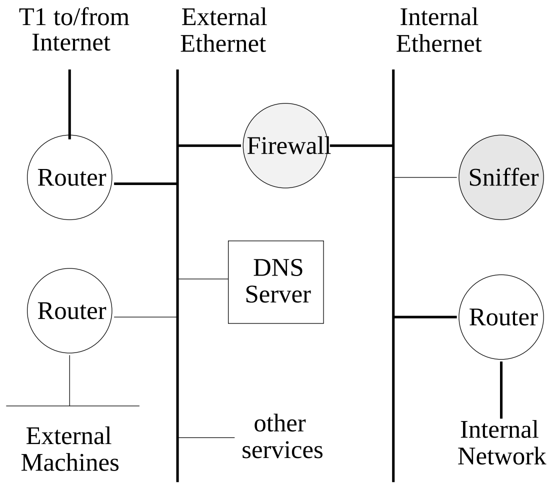
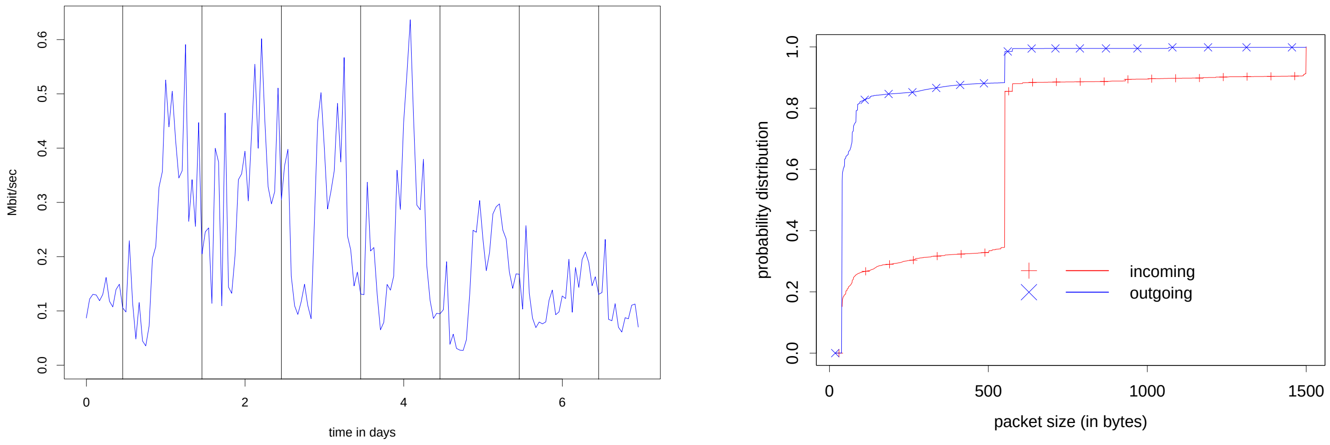


Figure 1: **Trace Collection Environment:** This figure shows the location of the packet monitor on the internal Ethernet that carries traffic behind the firewall to and from the external Internet.

The Ethernet segment carries Internet traffic for various types of end-point machines, including office PCs, Sun workstations, shared compute servers, proxy servers, and modem-connected hosts. Since user behavior may vary based on the computing platform, we evaluate the flow statistics for each of these types of hosts on the Ethernet segment. When possible, local IP addresses are classified automatically based on information available in the packet-level traces. In particular, Web request messages include an optional user-agent field that identifies the operating system running at the client site. Based on the IP address and the client operating system, we classify Linux and/or Windows systems as personal computers. Any IP address associated with multiple types of machines (e.g., Windows and SunOS) is classified as a proxy server. Finally, modem traffic is identified by a set of IP addresses dedicated to a local dial-up service. In the absence of sufficient information in the packet traces, a small number of remaining IP addresses were resolved manually. For example, a number of SunOS and IRIX systems were classified as shared compute servers based on local knowledge of these machines.

2.2 Traffic Characteristics

In this paper, we focus on a continuous one-week trace starting at 11am on Sunday February 16, 1997. Figure 2(a)



(a) Link utilization across time

(b) Cumulative distribution of packet sizes

Figure 2: **Traffic Characteristics:** This figure highlights the basic traffic characteristics of a one-week trace, starting at 11am on a Sunday. The left graph plots hourly averages of the throughput of the Ethernet segment with a vertical bar signifying midnight on each day, while the right graph plots the packet size distribution for incoming and outgoing traffic.

plots hourly averages of the bandwidth consumed on the Ethernet segment across the duration of the trace. The trace shows daily fluctuations, with the heaviest load during the work day, and smaller spikes in the evening hours; each weekday also shows a brief dip in network utilization during the lunch hour. Hourly averages of throughput remain below 0.6 megabits/second. However, much higher rates occur on a smaller time scale; for example, the Ethernet segment sustains throughputs of up to 3 megabits/second over one-minute intervals. The incoming and outgoing Internet traffic consists of a mixture of different packet sizes, as shown by the graph in Figure 2(b), which plots the probability that a packet has up to x bytes. Half of the outgoing traffic consists of 40-byte TCP acknowledgment packets, contributing to a low average packet size of 123 bytes; the remaining short packets are mainly HTTP requests. In contrast, less than one-fifth of the incoming packets stem from 40-byte TCP acknowledgments. In fact, more than 60% of the incoming packets have 552, 576, or 1500 bytes, which correspond to common maximum transfer unit sizes on the Internet.

Most of the incoming traffic stems from response messages from HTTP and FTP servers, as shown by the statistics in Table 1. Web response messages on the well-known IP port 80 contribute over half of the incoming bytes. By applying the `tcpreduce` tool [28], we estimate that FTP-data transfers are responsible for over one-third of the transfers on "unknown" ports; the remaining traffic on unknown ports appear to stem mainly from real audio and other HTTP-originated transfers. Table 1 also includes the average flow sizes in packets and seconds, where the flow duration does not include the 60-second timeout value for these port-to-port flows. The table shows that short request-response protocols, such as the network time protocol, generate extremely short-lived flows[1]. In general, with port-to-port flows and a 60-second timeout, a flow typically corresponds to a single TCP session. Still, periods of inactivity may cause the packets in a single TCP session to generate multiple consecutive flows. With the dramatic increase in the amount of Web traffic in the Internet, we focus the remainder of our study on HTTP flows. Despite the relatively low *average* size of Web flows, the heavy tail of the flow-size distributions and the ability to aggregate multiple transfers substantially reduce the overheads for shortcutting HTTP traffic, as shown in the next section.

3 HTTP Response Flows

In this section, we investigate the dynamics of HTTP response traffic by characterizing the flow-size distributions as a function of traffic type, end-point aggregation, and timeout value. The results show that certain protocol features and user/application conventions have significant influence on the network's ability to establish efficient shortcut connections for Web response traffic. To investigate these diverse application characteristics, we consider the probability distribution of flow sizes for a full week of trace data.

3.1 Interpreting Flow-Size Distributions

The diversity of the flow characteristics complicates the effort to capture the key properties in a single probability

[1] Interestingly, DNS flows have an average of 11 packets over 80.3 seconds, in contrast to the short DNS flows we see for internal traffic; the longer flows stem from communication between pairs of DNS servers, rather than isolated DNS responses to internal hosts.

Protocol	Percent bytes	Percent packets	Percent flows	Packets/ flow	Bytes/ packet	Seconds/ flow
http	52.33	42.50	73.38	16	612	5.3
smtp	3.28	6.49	8.20	22	251	6.0
dns	0.75	3.01	7.32	11	124	80.3
telnet	0.57	4.52	1.26	98	63	68.9
x11	0.26	1.98	0.46	118	66	249.3
ftp-cntrl	0.07	0.36	0.95	10	95	32.4
ntp	0.02	0.10	2.72	1	76	0.4
other	0.38	0.87	0.68	35	220	28.7
unknown	42.32	40.17	5.03	219	523	26.5

Table 1: **Incoming Internet Traffic by Protocol:** This table summarizes the characteristics of incoming traffic for each well-known port that contributes at least 0.5% of the bytes, packets, or flows, for port-to-port flows with a 60-second timeout. The rest of the well-known ports are classified as "other," while the remaining "unknown" ports predominantly handle FTP-data, HTTP, and real-audio transfers.

distribution graph. As an initial characterization of the one-week trace, Figure 3 plots the distribution of the number of bytes per flow for incoming traffic, with port-to-port flows and a 60-second timeout, in three different formats. To focus on the trends across a wide range of flow sizes, each graph plots the probability distributions with a logarithmic scale on the x-axis; the y-axis remains on a linear scale to emphasize the most common flow sizes, although a logarithmic scale on the y-axis would better highlight the heavy tail of the distribution. The cumulative distribution function in Figure 3(a) shows the probability that a flow has up to x bytes. Although Figure 3(a) shows that the traffic has a relatively broad mixture of flows between 100 and 100000 bytes, the graph does not clearly emphasize the most common flow sizes. By plotting the *derivative* of the cumulative distribution, Figure 3(b) has peaks that correspond to a significant portion of flows within a certain range of sizes, as in a histogram.

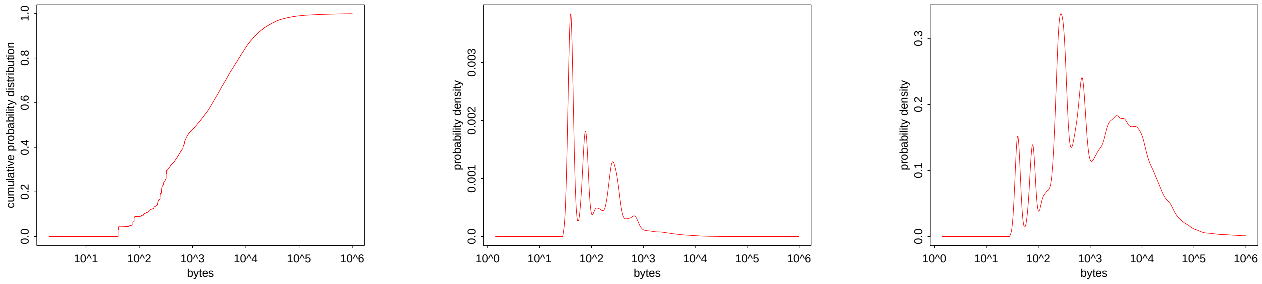
For example, the traffic mix includes a large number of flows with around 250–300 bytes. These trends become more apparent in Figure 3(c), which plots the probability density of the *logarithm* of the flow size. Coupled with a logarithmic scale on the x-axis, plotting the density of the logarithm of the data facilitates direct comparisons between different parts of the graphs based on the area under the curve. For example, approximately one-third of the flows have between 1000–10000 bytes, while another one-third have between 100–1000 bytes. This trend is only visible when plotting the density of the logarithm of the flow sizes. Also, the plot in Figure 3(c) better highlights the shape in the tail of the distribution, though the heavy tail of flows with over one million bytes is still not very noticeable. Throughout the remainder of this section, the flow-size graphs follow the convention in Figure 3(c). Where relevant, we also describe the tail of the distribution, since these long-lived flows have a significant impact on the performance of shortcut techniques.

3.2 Unique Characteristics of Web Traffic

To highlight the unique characteristics of HTTP traffic, Figure 4 graphs the distribution of flow sizes for incoming HTTP, Telnet, and SMTP packets. Each flow corresponds to packets with the same IP addresses and port numbers at both end-points and a 60-second timeout. Figure 4(a) plots the total number of bytes in the flow, whereas Figure 4(b) plots the flow duration, measured as the time between the first and last incoming packet; that is, the flow duration does not include the 60-second timeout value. By separating the flows by protocol, these plots highlight the unique application characteristics that generate the many peaks in Figure 4. Also, all three protocols have a fair number of short flows that typically correspond to failed service requests or occasional cases where consecutive packets arrive more than one minute apart.

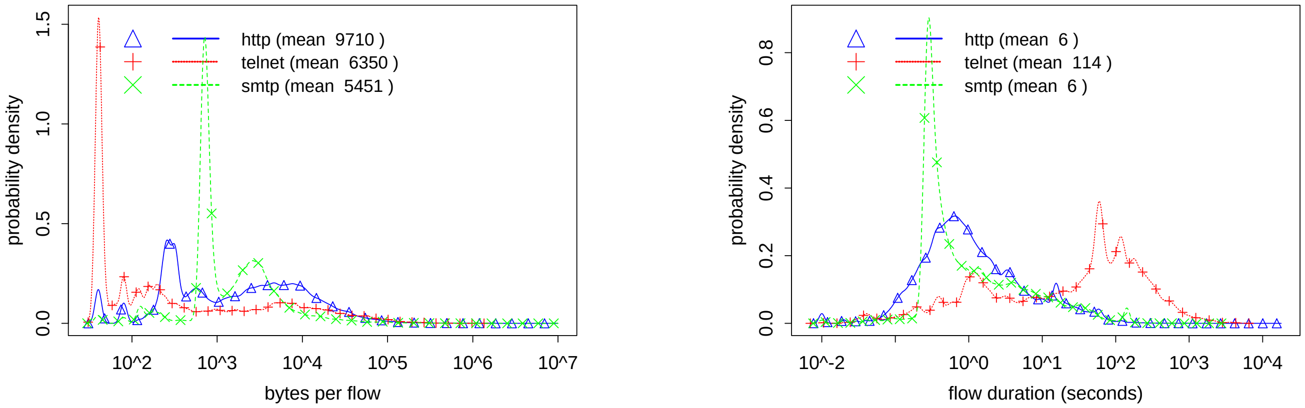
As shown in Table 1, Telnet flows typically have the longest durations, even though 60-second periods of inactivity can split a single telnet session into multiple consecutive flows. Despite the relatively long duration of Telnet flows, these interactive flows do not generate as many bytes as the HTTP flows. SMTP traffic has a high concentration of flows with just under 1000 bytes. These flows stem from the minimum overhead for transmitting an e-mail message, while the remaining flows, centered at 3000 bytes, consist of longer messages. The HTTP responses include numerous flows with a large amount of data, as shown by the heavy tail in Figure 4; these results are consistent with recent characterizations of Web document sizes [3, 12].

With the relatively large timeout value, a port-to-port HTTP flow typically corresponds to a TCP session for a single transfer from a server to a client. The HTTP flow sizes in Figure 4 fall into three main categories. The first region consists of flows with less than 150 bytes, stemming from failed TCP sessions and HTTP error messages. In the second region, 24% of the flows have between 150 and 300 bytes. Many of these transfers involve small response messages that indicate that the client can use its cached copy of a Web page, although some small Web pages fall



(a) Cumulative distribution (b) Probability density (c) Probability density of log

Figure 3: **Comparing Probability Distributions:** These graphs plot the probability distribution of flow sizes in bytes. The distributions consider port-to-port flows of incoming traffic, with a 60-second timeout.



(a) Length in bytes (b) Duration in seconds

Figure 4: **Flow Sizes for Different Protocols:** These graphs plot the probability density function of the logarithm of flow sizes for incoming HTTP, Telnet, and SMTP packets, for port-to-port flows with a 60-second timeout.

in the same size range. Finally, the remaining 69.5% of flows correspond to the actual transfer of Web data from the server to the client. Inspecting the HTTP response headers in the packet-level data verifies these trends: 72% of responses were "okay" retrievals, 18% were "not modified" messages, and 10% were various error messages (e.g., "no content," "moved temporarily," and "not found").

3.3 HTTP Flows by Machine Type

To further dissect the HTTP response traffic, Table 2 shows the average flow sizes for four different types of client machines, for port-to-port flows with a 60-second timeout. The end-point machines have unique flow characteristics due to a variety of possible factors, including network access rate, machine speed, Web cache size, and user demographics. Modem-connected hosts and personal computers have nearly the same distribution of the number of bytes and packets per flow. However, the low modem access speed inflates the flow duration; this is particularly apparent in Figure 5, where modem-connected hosts have a much higher minimum flow duration than the other machine types. These long flows translate directly into longer-lived shortcut connections to transfer the same amount of data. In contrast, the flows to shared UNIX machines typically have more bytes and packets, with the larger average stemming mainly from a heavier tail; these longer transfers may stem from large Postscript and image files.

Finally, traffic to proxy servers tended to have a more bimodal distribution, with a mixture of shorter and longer flows. The proxy machines typically have large, shared Web caches that result in an larger number of cache-hit messages, with small byte counts and fast transfer times. After removing the cache-hit messages, the remaining proxy traffic has a disproportionate number of large HTTP transfers, including Postscript and image files, similar to the traffic to UNIX compute servers. These large transfers typically experience higher throughput than small trans-

Machine Type	Bytes	Packets	Seconds
Modem host	8317	12.1	9.3
Proxy server	9206	18.2	2.8
Compute server	12136	21.5	6.8
Personal computer	8779	11.9	5.2

Table 2: **Average HTTP Flow Sizes for Different Client Machines:** This figure shows the average flow sizes for incoming HTTP traffic across four types of client machines, for port-to-port flows with a 60-second timeout.

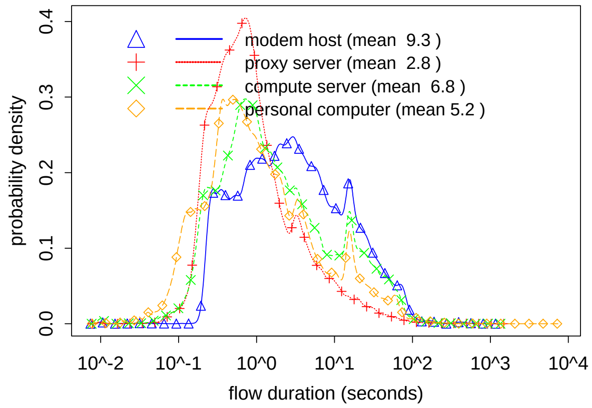


Figure 5: **Flow Duration Distribution for Different Client Machines:** This graph plots the probability density function of the logarithm of flow durations for four types of client machines, for port-to-port flows with a 60-second timeout.

fers, since the TCP window grows as the sender progresses through slow-start. As a result, these long transfers experience lower delay relative to the flow size. High throughput for the long flows, coupled with low latency for short cache-hit messages, may partially explain the short average flow durations for HTTP response traffic to the proxy servers. Although the various types of end-point machines have important effects on the flow-size distributions, we focus below on the fundamental properties that stem from HTTP conventions.

3.4 Combining Multiple Web Responses

To further characterize the dynamics of HTTP traffic, Figure 6 graphs the distribution of flow sizes for three different levels of end-point aggregation. At the lowest level, the port-to-port curves correspond to packets with the same IP addresses and port numbers at both end-points, effectively repeating the HTTP response results from Figure 4. By ignoring the port number at the two end-points, a single flow at the host-to-host level can include HTTP packets from different TCP sessions from the same Web server to the same Web client. Host-level aggregation flattens the peak in Figure 6(a) by combining consecutive transfers into a single flow. Since host-to-host aggregation decreases the total number of flows, the short single-packet flows represent a larger portion of the distribution, as seen by the slightly higher peaks at the far left in both Figure 6(a) and Figure 6(b).

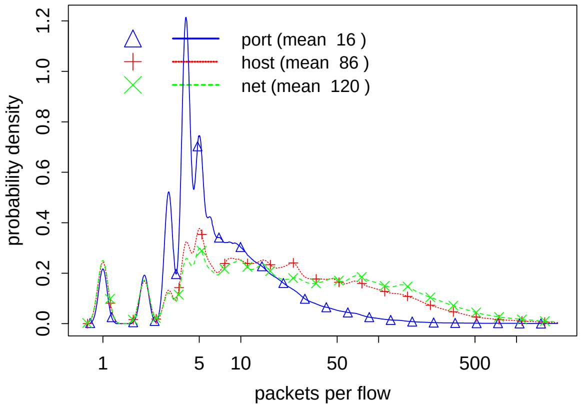
For the larger flows, the host-to-host curve in Figure 6(b) closely resembles the port-to-port curve, except for a shift to the right by a factor of four. This can be partially attributed to the Web browser convention of opening four simultaneous TCP sessions for transmitting the inline contents within a Web page. Opening multiple TCP sessions allows the client to pipeline HTTP requests and increases throughput by circumventing the effects of TCP flow control. By aggregating traffic at the host-to-host level, the network can counteract the potential unfairness of this policy. In particular, the network could assign the same bandwidth to each HTTP shortcut connection. Enforcing this fair allocation, through traffic shaping or link scheduling in the network, can help ensure that aggressive clients cannot degrade the performance of other users by opening multiple TCP sessions to the server.

In addition to combining *concurrent* transfers at the Web server, host-to-host flows can capture *consecutive* accesses by the same client. With a 60-second timeout value, host-level aggregation can combine several transfers from the same Web server into a single flow. Compared to port-to-port plots, host-to-host aggregation shifts much of the area from 4–10 packets and 250–10,000 bytes to 30–500 packets and 10,000–1,000,000 bytes. As more browsers and servers begin to use persistent TCP sessions that span multiple Web accesses [26], even port-to-port flows could include multiple Web transfers. In fact, the results in Figure 6(a) provide an initial indication of the potential performance benefits of this application-level aggregation. A larger flow timeout parameter increases the likelihood of capturing consecutive HTTP transfers between the same two endpoints. Experiments in Section 4 with different timeout values can guide policies for when the network should close idle shortcut connection (or, similarly, when a server should close a persistent TCP session).

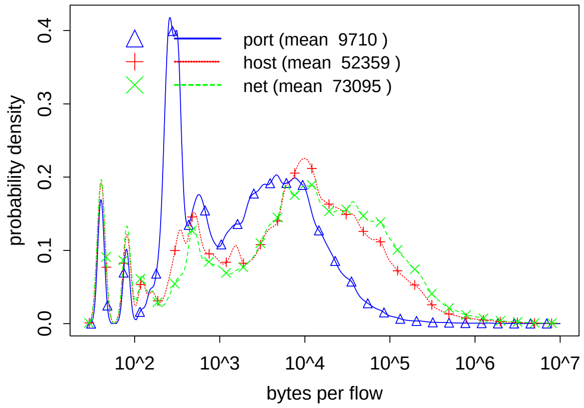
Aggregating traffic beyond the host level does not substantially change the flow length distributions, as shown in Figure 6. We define a *net* to include all hosts that share the first 16 bits of their IP addresses. Aggregation at the net-to-net level does not have a significant impact on flow sizes in our trace, as shown in Figure 6. Instead, reducing the network overheads for transporting HTTP responses requires more aggressive techniques for detecting long-lived flows, as discussed in Section 4, and for aggregating traffic over smaller portions of the route between the servers and the clients, as discussed in and Section 5.

4 Network Shortcut Overheads

The flow-size distributions give an initial indication of the network resources necessary to establish and maintain short-



(a) Length in packets



(b) Length in bytes

Figure 6: **Flow Sizes for Web Responses:** These graphs plot the probability density function of the logarithm of flow sizes for Web responses for four levels of aggregation, with a 60-second timeout.

cut connections for long-lived flows. After a brief evaluation of how packet triggers affect the amount of shortcut traffic, this section studies how triggers, timeouts, and end-point aggregation affect the signaling and switching overheads in the network on a variety of timescales. By evaluating these performance metrics on a variety of timescales, we can better characterize the processor and switch resource requirements for shortcutting long-lived flows. The results show that it is possible to significantly reduce the peak signaling load by occasionally introducing a modest delay in migrating flows to shortcut connections. On a longer time scale, time-of-day load fluctuations suggest the use of dynamic shortcutting schemes that adjust the trigger, timeout, or aggregation policy a few times a day.

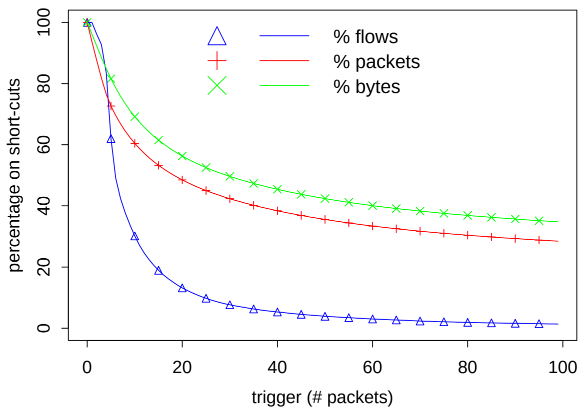
4.1 Proportion of Shortcut Traffic

After detecting a flow, based on the end-point addresses and a timeout value, the network must decide if and when to create a shortcut connection. Establishing a shortcut reduces the forwarding load on the default path, at the expense of signaling and maintaining a new connection. To quantify these tradeoffs, Figure 7 plots the amount of traffic that travels on a shortcut connection as a function of the triggering policy. The percentage of shortcut flows in Figure 7 corresponds to the complementary cumulative distribution of the number of packets per flows. The steep nature of the curve stems from the large number of short-lived flows in Figure 6(a). By waiting until a flow has transferred at least 10–20 packets, the network can avoid establishing shortcut connections for the large number of short-lived flows generated by control messages, cache hits, and small Web transfers. These basic results are consistent with the findings of previous Internet traffic studies [11, 22, 25, 32].

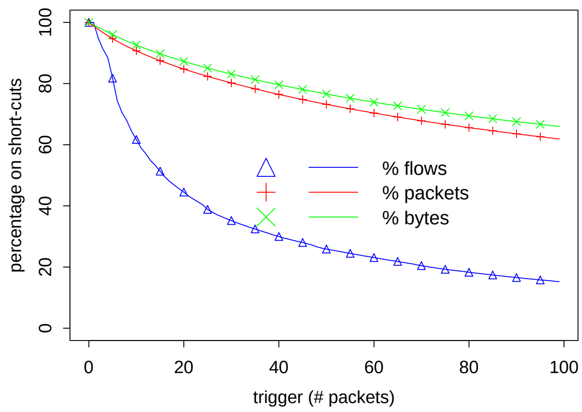
In addition to removing the large number of short-lived flows, the triggering policy also forces the first x packets of each long-lived flow to travel on the default path. Still, the heavy tail of the flow-size distributions in Figure 6 ensures that the network can forward a large proportion of the packets and bytes along shortcut connections. For example, in Figure 7(a), a 25-packet trigger filters over 90% of the port-to-port flows, while still permitting 45% of the packets and 53% of the bytes to travel along shortcuts. The slightly higher proportion of bytes stems from the large packets in long HTTP responses. Comparing Figure 7(a) and Figure 7(b), an x-packet trigger is somewhat less effective in reducing the proportion of shortcuts for host-to-host flows, since the coarser level of aggregation increases the number of packets in each flow. For example, applying a 25-packet trigger under host-level aggregation removes only 61% of the flows; as a result, shortcut connections carry 82% of the packets and 85% of the bytes.

4.2 Shortcut Setup Rate

The trends in Figure 7 translate directly into reductions in the setup rate for establishing shortcut connections. However, the setup rate can fluctuate dramatically across time, complicating the effort to provision signaling resources in the network. Experiments with the one-week packet trace show that the setup rate typically varies in proportion to the traffic load. To quantify these variations across time, Figure 8(a) plots the maximum shortcut setup rate over a variety of time scales, ranging from 10 seconds to nearly 3 hours. The graph plots the setup rate for four different packet triggers (0, 5, 10, and 50) for port-to-port flows with a 60-second timeout value. To generate the graph, we determine the number of shortcuts triggered in each interval, where the interval size is a multiple of 10 seconds. In

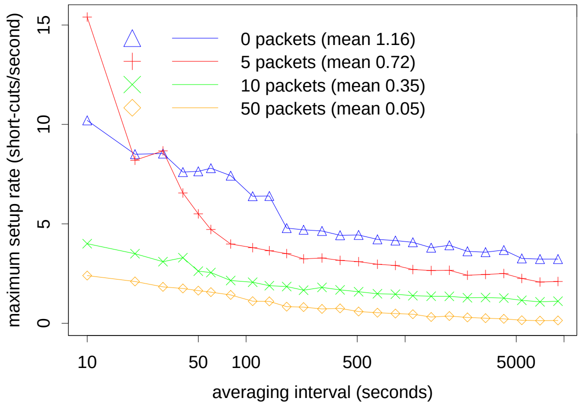


(a) Port-to-port flows

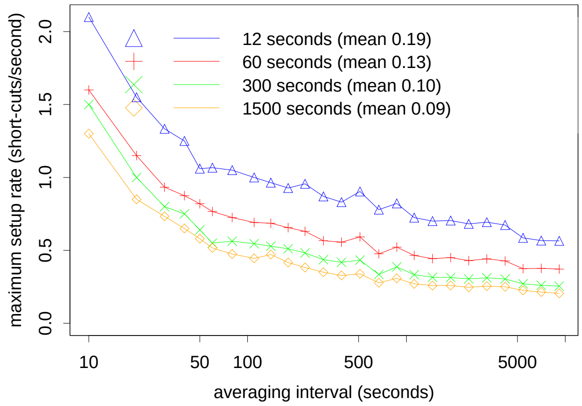


(b) Host-to-host flows

Figure 7: **Percent Shortcut Traffic:** These graphs plot the percentage of bytes, packets, and flows that travel on a shortcut connection as a function of the packet-count trigger, for HTTP response traffic with a 60-second timeout.



(a) Packet trigger (port-to-port flows)



(b) Timeout value (host-to-host flows)

Figure 8: **Shortcut Setup Rate on Different Time Scales:** These graphs plot the maximum setup rate for new shortcut connections across a range of time scales. Graph (a) compares four different packet triggers under a 60-second timeout and port-to-port flows, while graph (b) compares four different timeout values under a 10-packet trigger and host-to-host flows. The legend shows the average setup rate across the entire one-week trace.

general, the setup rate decreases as a function of the interval size, although small increases occasionally occur due to the use of non-overlapping time intervals. A sliding-window computation could conceivably remove this minor effect, at the expense of a significant increase in computational complexity. The trigger value determines exactly when a shortcut connection is established. Occasionally, smaller trigger values have larger peak setup rates, if the number of shortcut establishments increase during a single time interval.

In Figure 8(a), the plot for a 0-packet trigger represents the overheads for establishing a shortcut for every flow; these baseline results also highlight the overheads for performing flow detection, even when the network employs a more conservative triggering policy. On a 10-second time scale, a 0-packet trigger introduces a maximum of 10.2 shortcuts/second to handle HTTP response traffic on a moderately-loaded T1 link. During peak day-time hours, the trace shows sustained *average* rates of 4–5 flows/second. Projecting to a higher-speed backbone network, the flow arrival rate could easily swamp the signaling resources in modern switches. Higher triggers and coarser flow aggregation can reduce this load. Compared to the port-level aggregation in Figure 8(a), host-to-host flows exhibit similar trends, with a factor of four smaller setup rates. To better characterize the burstiness of the flow-arrival process, we also measured the degree of self-similarity of one-hour segments of the trace using wavelet techniques [15]. Counting the number of flows that arrive within each 100-millisecond time period in a single hour, we computed a Hurst parameter of 0.72 for port-to-port flows and 0.6 for host-to-host flows, suggesting that the flow-arrival process is indeed self-similar. The estimates of the Hurst parameter do not change significantly with the application of a 10-packet trigger (0.70 and 0.57 for port-to-port and host-to-host flows, respectively). The parameters drop to 0.62 and 0.50 for a 50-packet trigger, suggesting that the arrival process for very long flows is not as bursty.

A combination of techniques may be necessary to reduce the signaling load to acceptable levels during times of peak network usage. To adjust to *short-term* load variations, the network can temporarily delay the establishment of shortcuts for flows, even if the packet trigger has been reached. For example, Figure 8(a) shows that a 10-packet trigger generates up to 4 shortcuts/second on a 10-second time scale, but this number drops to 2 shortcuts/second on a 100-second interval. Instead of provisioning for the higher load, the network could allocate processing resources for establishing 2 shortcuts per second. During brief periods of overload, the signaling processor can queue shortcut requests and continue to forward packets on the default path. The results in Figure 8(a) suggest that this would not substantially delay the creation of shortcut connections. Also, the processor can avoid establishing shortcuts for any flows that terminate in the meantime.

To adjust to *longer* periods of heavy load, the network may need to relax the flow definition to apply coarser end-point aggregation and larger timeout values. Figure 8(b) experiments with different flow timeout values under host-level aggregation and a 10-packet trigger. Larger timeout values decrease the shortcut setup rate by capitalizing on the possibility of subsequent Web transfers between the same end-points. Compared to increasing the packet trigger, a larger timeout value can decrease the setup rate without forcing more packets to follow the default path; in fact, by combining more consecutive Web transfers in a single flow, large timeout values increase the number of packets that travel on the shortcut connection. However, timeout values in excess of five minutes offer diminishing returns [8, 11], due to the decreasing likelihood of subsequent transfers from the same Web server.

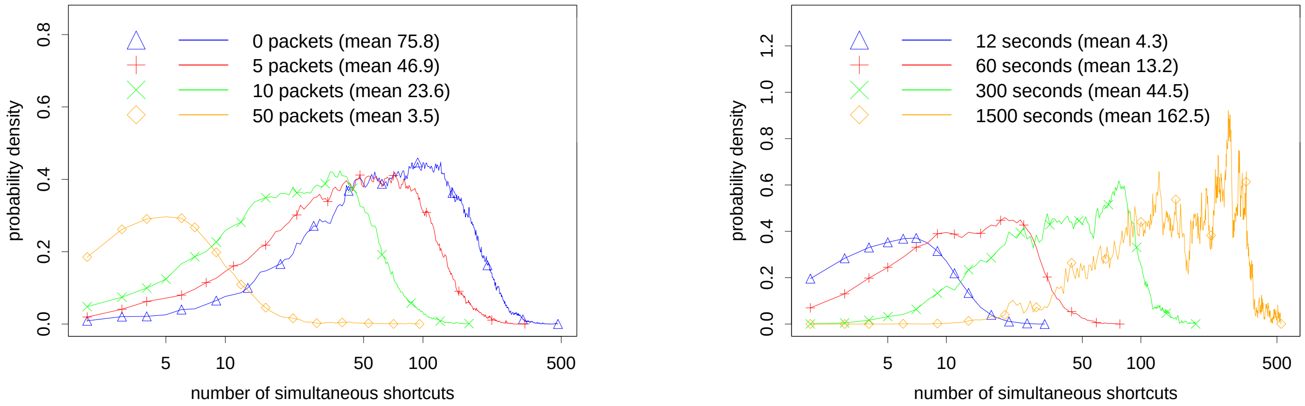
4.3 Simultaneous Shortcut Connections

In addition to affecting the signaling load, the timeout, trigger, and aggregation policies also influence the number of active shortcut connections across time. Depending on the underlying switching technology, the number of connections may impose a tighter restriction than the setup rate. Similar to the setup rate, the number of simultaneous shortcuts typically fluctuates in proportion to the link utilization, plotted in Figure 2(a). To focus directly on switch overhead, Figure 9(a) plots the probability density function of the number of shortcuts for four different trigger values, for port-to-port flows with a 60-second timeout. Moving to the right from a point on the x-axis, the area under the curve represents the proportion of time when the network must support at least x simultaneous connections. The legend indicates the average number of shortcut connections across the one-week trace, while the right side of each curve corresponds to the maximum value.

A 0-packet trigger introduces an average of 75.8 shortcuts, with sustained periods of more than 300 connections during peak day-time hours. In Figure 9(a), the curve for a 0-packet trigger serves as an upper bound for the other triggering policies, while also estimating the size of data structures necessary for the network to track active flows. A 10-packet trigger reduces the average number of shortcuts by a factor of 3.3 (from 75.8 to 23.6), while host-level aggregation reduces it by a factor of 3.8 (to 19.8); together, a 10-packet trigger and host-level aggregation reduce the average by a factor of 5.8 (to 13.2). Despite the large average and peak values for the 0-packet trigger, Figure 9(a) also shows a large portion of time with a fairly small number of connections. On the other hand, the absolute number of simultaneous shortcuts established by a particular policy will scale up with link speed and utilization, so that the number of shortcuts on a high-speed backbone link may exceed the capacity of current switches.

The above results illustrate that the ideal trigger, timeout, and aggregation options change with the time of day, suggesting the use of dynamic policies that vary across the day. For example, the one-week trace has three basic load periods corresponding to heavy load during the business day, moderate load on weekend days, and light load late at night, typical of a corporate networking environment; other parts of the Internet, such as residential access lines and backbone links, are likely to have different patterns. Although the network can support fine-grain flows with small trigger values during periods of lighter load, coarser aggregation and larger triggers may be necessary for stable operation during heavy load. However, an aggressive trigger reduces network overhead by forcing more packets to travel on the default-routed path, without enjoying the potential performance benefits of a shortcut route. Alternatively, the network could reduce signaling load by employing a larger timeout value, at the expense of increasing the number of active shortcuts. Since the network does not close a shortcut connection until the timeout expires, a large timeout value can substantially increase the number of shortcut connections, as shown in Figure 9(b). The network can limit this effect by terminating inactive shortcuts as the available connection resources are depleted.

Depending on the network configuration, the establishment of shortcuts may be limited by either the number of connections or the signaling capacity of the switch, or both; in fact, the appropriate balance of timeout and triggering policies may vary with the time of day. Although time-



(a) Packet trigger (port-to-port flows)

(b) Timeout value (host-to-host flows)

Figure 9: **Number of Simultaneous Shortcuts:** These graphs plot the probability density of the number of simultaneous shortcut connections across the one-week trace. Graph (a) compares four different packet triggers under a 60-second timeout and port-to-port flows, while graph (b) compares four different timeout values under a 10-packet trigger and host-to-host flows.

outs, triggers, and end-point aggregation can substantially reduce network overhead, each technique offers diminishing returns beyond a certain point. In addition, overuse of these techniques can degrade application performance. For example, a large trigger forces more packets to travel on the default path, without enjoying the performance benefits of a shortcut connection. Also, aggregating traffic beyond the host level can introduce unfair interactions between users that must share a single shortcut. In the next section, we consider new techniques that reduce network overhead without requiring unrelated users to share a shortcut connection.

4.4 Flow vs. Packet Measurements

Generating the distributions in Figure 8 and Figure 9 required fairly time-consuming computations across a large packet trace. To reduce the complexity of evaluating shortcutting policies, or to study parts of the network that cannot collect packet-level statistics, we would prefer to operate on flow-level traces. For example, these traces could include the start and finish time of each flow, as well as the address and port information and the total number of packets and bytes. Although such traces could be used to determine the flow-size distributions in Section 3, the shortcutting overheads depend on the spacing of packets within a flow. In working with flow-level traces, we can only estimate the time when a shortcut is triggered, based on an approximation of packet arrival times. One plausible approach is to assume a uniform spacing of equal-sized packets throughout the flow duration; for an n-packet, d-second flow starting at time t, we estimate that the x-th packet arrives at time $t + dx/(n-1)$, $x = 0, 1, \ldots, n-1$.

To evaluate the impact of this simplifying assumption, we compared the shortcut metrics obtained from uniform packet spacing to our actual results from using per-packet timestamps. The distributions of the number of shortcut connections from the flow-level statistics were virtually indistinguishable from the plots in Figure 9. Still, the flow-level traces slightly underestimate the average number of simultaneous shortcuts (by less than 2.5%) for all timeout values larger than 12 seconds. We speculate that the small, but consistent, underestimation stems from the use of parallel TCP sessions to download embedded images of different sizes. In the packet-level traces, these flows generate shortcut connections that start at almost exactly the same time, whereas the assumption of uniform packet spacing can result in different triggering times in the flow-level traces. In contrast, the average shortcut setup rate is the same for the packet-level and flow-level statistics, since this metric depends only on the distribution of the number of packets in a flow.

Despite having the same mean setup rate, the worst-case setup rate differs between the packet-level and flow-level traces, particularly on the small timescales. This is not surprising, since packet spacing has a significant influence on the number of shortcuts that arrive in any small interval of time (say, 10–100 milliseconds). In general, packet-level dynamics cannot be ignored in understanding these small timescale effects [14]. Yet, understanding the flow dynamics at a coarser granularity may be sufficient for provisioning network resources for creating and maintaining shortcut connections. The flow-size distributions, combined with accurate measurements of link utilization (or, alternatively, the sequence of flow-arrival times), can be used to estimate the resource requirements. Then, additional resources could be allocated to account for variability on small timescales. Finally, during transient periods of particularly heavy load, the network can temporarily delay the establishment of shortcut connections by continuing to forward packets on their default routes.

5 Route Aggregation

High-speed backbone networks may require more aggressive aggregation techniques to reduce signaling and switch overheads below the levels in Section 4. In this section, we propose a flow definition that generalizes the concept of end-point addresses to consider a *subset* of the path between the source and destination hosts. By drawing on this type of routing information, the shortcutting mechanism can group packets that have a common path through a portion of the network, such as the domain of a single service provider. A similar notion of partial routes has also been proposed in the Nimrod architecture to improve the scalability of routing computations [9]. Drawing on traceroutes to the Web servers in our packet-level trace, our preliminary results show that combining traffic on partial routes can further reduce the setup rate for shortcut connections.

5.1 Traffic Flows Along Partial Routes

The previous sections consider a flow model based on the IP address and port number at both the source and destination hosts. For the HTTP response traffic in our trace, the source addresses come from a diverse collection of Web servers that span across the Internet. As discussed in Section 3.4, aggregating this HTTP response traffic beyond the host level does not substantially change the flow-size distributions, since the traces do not show significant subnet and net locality on the time scale of a 60-second timeout (see Figure 6). Although some links in the Internet carry packets for a small number of subnet and net addresses, the ability to aggregate traffic at the subnet and net levels degrades with the distance that packets must travel through the network. In a large network, a shortcut connection would typically carry traffic from just a single server to a single client (or small set of clients). Hence, whenever a client accesses a different Web server, the network would define a new flow and, ultimately, a new shortcut connection for the response traffic. Similarly, whenever a Web server initiates a response to a different client, the network would have to incur the overheads for establishing a new shortcut connection.

As an initial characterization of the diversity of routes, we performed traceroutes to each Web server in our one-week packet trace. To avoid mixing these UDP probes and ICMP replies with the traffic in the packet-level traces, the traceroutes were run just after the week of data collection completed. Although the traceroute results do not necessarily indicate the actual routes taken by the HTTP requests, or the reverse paths followed by the corresponding responses, they do give a basic indication of the fanout of traffic as it crosses the network. Also, recent studies indicate that a majority of routes are stable for periods of days or weeks [29]. The traceroutes from AT&T Research show 26 different "first hops" toward the 23060 different server IP addresses in the one-week trace. The servers have 14390 different net addresses, with 13803 having successful traceroutes. As hop-count increases, these numbers grow in an exponential sequence; for example, the first seven hops have 26, 71, 137, 267, 406, 916, and 1508 different outcomes, respectively.

Based on these traceroute results, we can model the creation of shortcut connections across networks of different sizes. As an initial approach, we consider spheres of locality around the AT&T Research T1 link, based on the traceroutes to the server sites. To evaluate a three-hop region, for example, we assume that each of the 137 IP routers acts as an ingress switch for the HTTP response traffic en route to the T1 link. In this context, each flow originates at one of these routers (instead of using the actual source IP address of the Web server) and terminates at the client site. Although our traceroute experiments actually follow the forward path from the client to the server, the model still highlights the basic performance trends for the last h hops of the route from the server to the client. Similarly, the model can also project the overheads of establishing shortcut connections from a Web server (or set of servers) along the first h hops towards its client sites.

5.2 Partial-Route Aggregation

Drawing on this model of partial routes, we evaluated the overheads for 3-hop and 7-hop shortcut connections. The first row of Table 3 shows the results for host-to-host flows with a 60-second timeout and a 10-packet trigger, using the Perl scripts and Splus functions described in Section 2 (except that the h-hop router's IP address now acts as the "source" address of the flow). The mean setup rate and number of shortcuts for *end-to-end* (∞) flows are slightly smaller than the average for a 60-second timeout in Figure 8(b) and Figure 9(b), since the experiments in Table 3 omit any flows that did not have a successful traceroute. Aggregating traffic along a portion of the route decreases both the setup rate and the number of simultaneous connections; 7-hop routes reduce the setup rate and number of shortcuts 20% and 11% respectively, while 3-hop routes reduce them by 37% and 16%. In addition, route aggregation increases the proportion of traffic that travels on shortcuts, for the same trigger and timeout values. The 7-hop policy places 94.0% of the bytes on shortcut connections, compared to just 92.2% without route aggregation; 3-hop routes achieve an even higher proportion, with over 96.0% of the bytes traveling on shortcuts.

Partial-route aggregation reduces network overheads, while increasing the proportion of shortcut traffic, by combining concurrent and consecutive HTTP transfers when one or more Web servers communicate with the same client. To belong to the same flow, these transfers must travel through a common h-hop router on the timescale of the

Address Aggregation	Flow Timeout	Setup Per Sec ∞	7	3	Num. Connections ∞	7	3	% Shortcutted ∞	7	3
host-to-host	60 seconds	0.124	0.099	0.078	12.18	10.78	10.19	92.2	94.0	96.0
host-to-host	300 seconds	0.097	0.068	0.040	41.46	32.98	25.13	94.0	95.9	97.9
host-to-net	60 seconds	0.121	0.068	0.008	12.07	9.16	3.52	92.4	95.9	99.6
host-to-net	300 seconds	0.091	0.038	0.002	40.26	22.65	4.52	94.3	97.7	99.9

Table 3: **Partial-Route Aggregation:** This table summarizes experiments with traffic aggregation along an h-hop portion of the route to the client, or set of clients. Aggregating along a portion of the route decreases the number of connections and the setup rate, while increasing the percentage of bytes on shortcuts.

60-second timeout. By focusing closely on specific flows in the trace, we find that partial-route aggregation combines transfers from replicas of the same Web site, as well as related servers at the same institution; quite often, these servers have the same net and subnet addresses. Aggregating across partial routes also combines transfers from different sites in the same web-hosting service, or other sites that happen to route through the same h-hop router. Compared to end-to-end flows based on the source and destination addresses, partial-route flows benefit more from larger timeout values, as shown by the second row in Table 3. While a 60-second timeout may not be long enough to capture the user "think time" between consecutive Web accesses, a 300-second timeout with partial-route aggregation allows a single shortcut to aggregate transfers from different Web servers to the same client. For example, the larger timeout value reduces the average setup rate by nearly a factor of 2 (from 0.078 to 0.040 per second) for the 3-hop aggregation policy.

Aggregating traffic along partial routes has an even larger benefit when a single shortcut connection can be shared by multiple Web clients. For example, a network may combine related users into a single flow end point, particularly when the Web clients belong to a single company or university. For the partial-route flow model and the AT&T Research trace, this effectively combines all of the users into one destination address, which receives traffic from a number of different h-hop ingress points. For the 7-hop configuration, with 1508 ingress routers, this reduces the average setup rate by 30% (from 0.099 to 0.068 shortcuts/second), beyond the results for host-to-host flows; the 3-hop setup rate experiences an even more dramatic reduction, falling by a factor of 9.8 (from 0.078 to 0.008). Similarly, the average number of simultaneous shortcuts falls by a factor of 15% and 65%, for the 7-hop and 3-hop routes respectively, compared to the results for host-to-host flows. A larger timeout value offers further reduction in the setup rate, at the expense of an increase in the number of simultaneous connections.

These initial results suggest that aggregating traffic along partial routes provides a desirable alternative to selecting coarse-grained end-to-end flows. Compared to flows between the source and destination end points, partial-route flows can reduce the shortcut setup rate without forcing unrelated clients to share the same shortcut. Since small network areas (lower h values) have lower overheads, large networks may benefit from dividing the network into multiple regions, with separate shortcut connections across each part of the route. Large networks are already likely to require at least a logical partitioning into different regions to support scalable addressing and routing for shortcut connections. Although partial-route flows may require packets from a single TCP session to traverse a sequence of shortcut connections that start and end inside the network, the reduction in the setup rate and number of connections in each region can play an important role in scaling to large network configurations.

6 Conclusions and Future Work

Shortcutting of long-lived traffic flows offers an efficient way to capitalize on recent advances in high-speed switching hardware. Since the World Wide Web has a dominant influence on network dynamics in the modern Internet, we have performed a detailed characterization of HTTP response traffic to evaluate the basic cost-performance tradeoffs in flow switching. A comparison with other types of traffic highlights the unique influence of the HTTP protocol and user behavior on the flow-size distributions and the benefits of end-point aggregation. Further division of the traffic by machine type shows how differences in network access rate, machine speed, cache size, and user demographics can affect the basic flow-size distributions. To characterize the network overhead for flow switching, we present the distribution of important metrics, including the percentage of traffic that follows shortcuts, the shortcut setup rate, and the number of simultaneous shortcuts, while varying the timeout, trigger, and aggregation policies. Finally, we evaluate new flow definitions that consider a subset of the path between the source and destination hosts.

Our results suggest several possible schemes for limiting the shortcut setup rate and number of simultaneous connections by temporarily delaying the creation of shortcuts during transient periods of heavy load. Since incoming packets can continue to follow the default path, the network has additional latitude to postpone establishment of shortcut connections. As part of future work, we plan to evaluate specific new policies that balance the short-term tradeoffs between processor and network load. These policies can extend existing schemes that delay connection setup requests in response to signaling failures [13,23]. Drawing on our experiments with partial-route aggregation, we are also investigating the policy and performance implications of combining traffic along a portion of the route, in lieu of employing end-to-end flows with coarser aggregation or larger triggers. Finally, to extend the traffic characterization work, we are studying a more detailed breakdown of Web traffic by content type, as well as the implications of push technology and the new features in the emerging HTTP 1.1 standard. These experiments should lend additional insight into the cost-performance tradeoffs of establishing shortcut connections for long-lived HTTP flows.

Acknowledgments

We would like to thank Albert Greenberg, Vikram Saksena, Nick Duffield, Chuck Kalmanek, Partho Mishra, and K.K. Ramakrishnan for their contributions to the traffic measurement effort. Thanks also to Kimberly Claffy and Balachander Krishnamurthy for helpful discussions on earlier versions of the paper.

Anja Feldmann is a member of the Networking and Distributed Systems Center at AT&T Labs – Research in Florham Park, New Jersey. Her current research interests include Internet measurement and traffic characterization, network performance debugging, and improving Web performance. She received a M.S. degree in Computer Science from the University of Paderborn, Paderborn, Germany, in 1990 and M.S. and Ph.D. dgrees in Computer Science from Carnegie Mellon University in Pittsburgh, USA, in 1991 and 1995, respectively. She can be reached via electronic mail to anja@research.att.com, and on the Web at http://www.research.att.com/~anja.

Jennifer Rexford is a member of the Networking and Distributed Systems Center at AT&T Labs – Research in Florham Park, New Jersey. Her research focuses on network measurement, routing protocols, and video streaming, with an emphasis on efficient network support for performance guarantees. She received a B.S.E. degree in electrical engineering from Princeton University in 1991, and M.S. and Ph.D. degrees in computer science and engineering from the University of Michigan in Ann Arbor, in 1993 and 1996, respectively. She can be reached via electronic mail at jrex@research.att.com, and on the Web at http://www.research.att.com/~jrex.

Ramón Cáceres is a Principal Technical Staff Member at AT&T Labs – Research in Florham Park, New Jersey. He received a B. Eng. in Electrical Engineering from McGill University, Canada, in 1983, and an M. S. and Ph. D. in Computer Science from the University of California at Berkeley in 1984 and 1992, respectively. Between 1985 and 1987 he developed TCP/IP-based networking products for Pyramid Technology, a Silicon Valley startup. Between 1992 and 1994 he pursued research in wireless networking and mobile computing at Matsushita Information Technology Lab in Princeton, New Jersey. He joined AT&T Bell Labs in 1994. His current research interests include network performance measurement, network traffic characterization, wireless networking, and mobile computing. He can be reached via electronic mail to ramon@research.att.com, and on the Web at http://www.research.att.com/~ramon.

References

[1] Arup Acharya, Rajiv Dighe, and Furquan Ansari. IP switching over fast ATM cell transport (IPSOFACTO): Switching multicast flows. In *Proc. IEEE GLOBECOM*, pages 1850–1854, 1997.

[2] M. Acharya and B. Bhalla. A flow model for computer network traffic using real-time measurements. In *Proc. Inter. Conference on Telecommunication Systems*, March 1994.

[3] Martin F. Arlitt and Carey L. Williamson. Internet web servers: Workload characterization and implications. *IEEE/ACM Trans. Networking*, 5(5):631–644, October 1997.

[4] ATM Forum MPOA Sub-Working Group. *Multi-Protocol over ATM Version 1.0 (AF-MPOA-0087.000)*, July 1997.

[5] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, Mark Stemm, and Randy H. Katz. TCP behavior of a busy Internet server: Analysis and improvements. In *Proc. IEEE INFOCOM*, April 1998.

[6] P. Barford and M. E. Crovella. Generating representative web workloads for network and server performance evaluation. In *Proc. ACM SIGMETRICS*, June 1998.

[7] E. Basturk, A. Birman, G. Delp, R. Guerin, R. Haas, S. Kamat, D. Kandlur, P. Pan, D. Pendarakis, V. Peris, R. Rajan, D. Saha, and D. Williams. Design and implementation of a QoS capable switch-router. Technical Report RC 20848, IBM Research Division, January 1997.

[8] R. Caceres, P. Danzig, S. Jamin, and D. Mitzel. Characteristics of wide-area TCP/IP conversations. In *Proc. ACM SIGCOMM*, pages 101–112, September 1991.

[9] I. Castineyra, N. Chiappa, and M. Steenstrup. The Nimrod routing architecture. Internet Request for Comments (RFC 1992), August 1996.

[10] Hao Che, San-Qi Li, and Arthur Lin. Adaptive resource management for flow-based IP/ATM hybrid switching systems. In *Proc. IEEE INFOCOM*, April 1998.

[11] Kimberly C. Claffy, Hans-Werner Braun, and George C. Polyzos. A parameterizable methodology for internet traffic flow profiling. *IEEE Journal on Selected Areas in Communications*, 13(8):1481–1494, October 1995.

[12] Mark E. Crovella and Azer Bestavros. Self-similarity in world wide web traffic: Evidence and causes. In

Proc. ACM SIGMETRICS, pages 160–169, May 1996.

[13] Anja Feldmann. Impact of non-poisson arrival sequences for call admision algorithms with and without delay. In *Proc. IEEE GLOBECOM*, pages 617–622, November 1996.

[14] Anja Feldmann, Anna Gilbert, and Walter Willinger. Data networks as cascades: Explaining the multifractal nature of internet WAN traffic. In *Proc. ACM SIGCOMM*, September 1998.

[15] Anja Feldmann, Anna C. Gilbert, Walter Willinger, and T.G. Kurtz. The changing nature of network traffic: Scaling phenomena. *ACM Computer Communication Review*, 28(2), April 1998.

[16] Henry J. Fowler and Will E. Leland. Local area network traffic characteristics, with implications for broadband network congestion management. *IEEE Journal on Selected Areas in Communications*, 9(7):1138–1149, September 1991.

[17] V. Jacobson, C. Leres, and S. McCanne. `tcpdump`, available at ftp://ftp.ee.lbl.gov, June 1989.

[18] Raj Jain. Packet trains – measurements and a new model for computer network traffic. *IEEE Journal on Selected Areas in Communications*, SAC-4(6):986–995, September 1986.

[19] Y. Katsube, K. Nagami, , Shigeo Matsuzawa, and H. Esaki. Internetworking based on cell switch router - architecture and protocol overview. *Proceedings of the IEEE*, 85(12):1998–2006, December 1997.

[20] Srinivasan Keshav, Carsten Lund, Steven Phillips, Nick Reingold, and Huzur Saran. An empirical evaluation of virtual circuit holding time policies in IP-over-ATM networks. *IEEE Journal on Selected Areas in Communications*, 13(8):1371–1382, October 1995.

[21] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Trans. Networking*, 2(1):1–15, February 1994.

[22] Steven Lin and Nick McKeown. A simulation study of IP switching. In *Proc. ACM SIGCOMM*, pages 15–24, September 1997.

[23] Danny J. Mitzel, Deborah Estrin, Scott Shenker, and Lixia Zhang. A study of reservation dynamics in integrated services packet networks. In *Proc. IEEE INFOCOM*, pages 871–879, April 1996.

[24] Jeffrey C. Mogul, Fred Douglis, Anja Feldmann, and Bala Krishnamurthy. Potential benefits of delta encoding and data compression for HTTP. In *Proc. ACM SIGCOMM*, pages 181–194, September 1997.

[25] Peter Newman, Greg Minshall, and Tom Lyon. IP switching: ATM under IP. *IEEE/ACM Trans. Networking*, 6(2):117–129, April 1998.

[26] Venkata N. Padmanabhan and Jeffrey C. Mogul. Improving HTTP latency. *Computer Networks and ISDN Systems*, 28(1/2):25–35, December 1995.

[27] Guru Parulkar, Douglas C. Schmidt, and Jonathan S. Turner. a^{I}t^{P}m: a strategy for integrating IP with ATM. In *Proc. ACM SIGCOMM*, pages 49–58, August/September 1995.

[28] Vern Paxson. `tcpreduce`, available at http://ita.ee.lbl.gov/html/contrib/tcp-reduce.html.

[29] Vern Paxson. End-to-end routing behavior in the Internet. *IEEE/ACM Trans. Networking*, 5(5):601–615, October 1997.

[30] Vern Paxson and Sally Floyd. Wide-area traffic: The failure of Poisson modeling. *IEEE/ACM Trans. Networking*, 3(3):226–255, June 1995.

[31] Yakov Rekhter, Bruce Davie, Eric Rosen, George Swallow, Dino Farinacci, and Dave Katz. Tag switching architecture overview. *Proceedings of the IEEE*, 85(12):1973–1983, December 1997.

[32] Kevin Thompson, Gregory J. Miller, and Rick Wilder. Wide-area internet traffic patterns and characteristics. *IEEE Network Magazine*, 11(6):10–23, November/December 1997.

[33] Arun Viswanathan, Nancy Feldman, Rick Boivie, and Rich Woundy. ARIS: Aggregate route-based IP switching. Internet Draft (draft-viswanathan-aris-overview-00.txt), work in progress, March 1997.