In IEEE/ACM Transactions on Networking, to appear, February 2009.

Drafting Behind Akamai: Inferring Network Conditions Based on CDN Redirections

Ao-Jan Su, David R. Choffnes, Aleksandar Kuzmanovic, and Fabián E. Bustamante
Department of Electrical Engineering & Computer Science
Northwestern University, Evanston, IL 60208, USA
Email: {ajsu,drchoffnes,akuzma,fabianb}@cs.northwestern.edu

***Abstract*— To enhance web browsing experiences, content distribution networks (CDNs) move web content "closer" to clients by caching copies of web objects on thousands of servers worldwide. Additionally, to minimize client download times, such systems perform extensive network and server measurements, and use them to redirect clients to different servers over short time scales. In this paper, we explore techniques for inferring and exploiting network measurements performed by the largest CDN, Akamai; our objective is to locate and utilize quality Internet paths *without* performing extensive path probing or monitoring.**

Our contributions are threefold. First, we conduct a broad measurement study of Akamai's CDN. We probe Akamai's network from 140 PlanetLab vantage points for two months. We find that Akamai redirection times, while slightly higher than advertised, are sufficiently low to be useful for network control. Second, we empirically show that Akamai redirections overwhelmingly correlate with *network* latencies on the paths between clients and the Akamai servers. Finally, we illustrate how large-scale overlay networks can exploit Akamai redirections to identify the best detouring nodes for one-hop source routing. Our research shows that in more than 50% of investigated scenarios, it is better to route through the nodes "recommended" by Akamai, than to use the direct paths. Because this is not the case for the rest of the scenarios, we develop low-overhead pruning algorithms that avoid Akamai-driven paths when they are not beneficial. Because these Akamai nodes are part of a closed system, we provide a method for mapping Akamai-recommended paths to those in a generic overlay and demonstrate that these one-hop paths indeed outperform direct ones.

***Index Terms*— Akamai, CDN, edge server, DNS, measurement reuse, one-hop source routing**

I. Introduction

Content delivery networks (CDNs) attempt to improve web performance by delivering content to end users from multiple, geographically dispersed servers located at the edge of the network [2]–[5]. Content providers contract with CDNs to host and distribute their content. Since most CDNs have servers in ISP points of presence, clients' requests can be dynamically forwarded to topologically proximate replicas. DNS redirection and URL rewriting are two of the commonly used techniques for directing client requests to a particular server [6], [7].

Beyond static information such as geographic location and network connectivity, most CDNs rely on network measurement subsystems to incorporate dynamic network information on replica selection and determine high-speed Internet paths over which to transfer content within the network [8]. In this paper, *we explore techniques for inferring and exploiting the network measurements performed by CDNs for the purpose of locating and utilizing quality Internet paths without performing extensive path probing or monitoring.*

We focus our efforts on the Akamai CDN, which is perhaps the most extensive distribution network in the world – claiming over 15,000 servers operating in 69 countries and 1,000 networks [2]. Without Akamai's CDN, highly popular web enterprises such as Yahoo, Amazon, or The New York Times would be unable to serve the gigabytes of data per second required by the images, Flash animations, and videos embedded in their web sites. Given the global nature of the Akamai network, it is clear that any viable information about network conditions collected by Akamai can be beneficial to other applications; in this paper, we demonstrate how it can improve performance for routing in large-scale overlay networks.

This paper explores (i) whether frequent client redirections generated by Akamai reveal *network* conditions over the paths between end-users and Akamai edge-servers, and (ii) how such information can be utilized by the broader Internet community. We expect the first hypothesis to hold true because Akamai utilizes extensive network and server measurements to minimize the latency perceived by end users [9]. Thus, if the load on Akamai edge servers were either low or uniform over long time scales (one of the main goals of CDNs in general), then Akamai client redirections would indeed imply viable network path-quality information.

For the second hypothesis, we consider the application of overlay routing. As long as an overlay network can map a subset of its nodes to Akamai edge servers, the clients of such an overlay could use Akamai redirections as viable indications regarding how to route their own traffic. Because the number of nodes in large-scale overlay networks is typically several orders of magnitude larger than the total number of Akamai servers, finding hosts that share networks with Akamai edge servers should not be difficult. Moreover, Akamai deploys its edge servers within ISPs' networks at no charge [10]. This greatly reduces ISPs' bandwidth expenses while increasing the number of potential overlay nodes that can map their positions to Akamai servers.

The incentive for a network to latch onto Akamai in the above way is to improve performance by using quality Internet

A subset of this work appears in the Proceedings of ACM Sigcomm '06 [1].

paths *without* extensively monitoring, probing, or measuring the paths among the overlay nodes. In this work, we do *not* implement such an overlay network. Instead, we demonstrate the feasibility of this approach by performing a large-scale measurement study.

We conduct our study over a period of approximately two months, using a testbed consisting of 140 PlanetLab (PL) nodes. We initially measure the number of Akamai servers seen by each PL node over long time scales for a given Akamai customer (*e.g.*, `Yahoo`). The surprising result is that nodes that are further away, in a networking sense, from the Akamai network are regularly served by *hundreds* of different servers on a daily basis. On the other hand, a moderate number of servers seen by a client (*e.g.*, 2) reveals close proximity between the two. However, because different Akamai servers often host content for different customers, we show that the vast majority of investigated PL nodes see a large number of servers (and paths), *e.g.*, over 50, for at least one of the Akamai customers.

We then measure the redirection dynamics for the Akamai CDN. While the updates are indeed frequent for the majority of the nodes, the inter-redirection times are much longer in certain parts of the world, *e.g.*, as large as 6 minutes in South America. Our subsequent experiments indicate that such large time scales are not useful for network control; we show that even random or round-robin redirections over shorter time-scales would work better. Regardless, we discover that the redirection times for the vast majority of nodes are sufficient to reveal network conditions.

To show that network conditions are the primary determinant of Akamai's redirection behavior, we concurrently measure the performance of the ten best Akamai nodes seen by each of the PL nodes. By pinging, instead of fetching web objects from servers, we effectively decouple the network from the server latency. Our results show that Akamai redirections strongly correlate to network conditions. For example, more than 70% of paths chosen by Akamai are among approximately the best 10% of measured paths.

To explore the potential benefits of Akamai-driven one-hop source routing, we measure the best single-hop and direct path between pairs of PL nodes. For a pair of PL nodes, we concurrently measure the ten best single-hop paths between the source and the destination, where the middle hop is a frequently updated Akamai edge server. Our results indicate that by following Akamai's updates, it is possible to avoid hot spots close to the source, thus significantly improving end-to-end performance. For example, in 25% of all investigated scenarios, Akamai-driven paths outperformed the direct paths. Moreover, 50% of the middle points *discovered* by Akamai show better performance than the direct path.

Not all Akamai paths will lead to lower latency than the direct alternative. For example, a direct path between two nodes in Brazil will always outperform any single-hop Akamai path, simply because the possible detouring point are in the US. Thus, we develop low-overhead pruning algorithms that consistently choose the best path from available Akamai-driven and direct paths. The question then becomes, how often does a client need to "double-check" to ensure that Akamai-driven paths are indeed faster than direct paths. We show that these techniques always lead to better performance than using the direct path, regardless of frequency, and that the frequency can be as low as once every two hours before a client's performance significantly declines. Thus, we show that this Akamai-driven routing has the potential to offer significant performance gains with a very small amount of network measurement.

Finally, we demonstrate the potential benefits of Akamai-driven routing for wide-area systems based on extensive measurements on BitTorrent peers. We perform remote DNS lookups on behalf of BitTorrent nodes and manage to associate (map) BitTorrent peers to Akamai edge servers. We then use these CDN-associated peers as the intermediate routing nodes to demonstrate the feasibility of CDN-driven detouring.

This paper is structured as follows. Section II discusses the details of the Akamai CDN relevant to this study. In Section III, we describe our experimental setup and present summary results from our large-scale measurement-based study. Section IV further analyzes the measured results to determine whether Akamai reveals network conditions through its edge-server selection. After showing that this is the case, we present and analyze a second measurement-based experiment designed to determine the effectiveness of Akamai-driven, one-hop source routing in Sections V and VI. We discuss our results and describe related work in Section VII. Section VIII presents our conclusions.

II. How Does Akamai Work?

In this section, we provide the necessary background to understand the context for the ensuing experiments. In general, for a web client to retrieve content for a web page, the first step is to use DNS to resolve the server-name portion of the content's URL into the address of a machine hosting it. If the web site uses a CDN, the content will be replicated at several hosts across the Internet. A popular way to direct clients to those replicas dynamically is DNS redirection. With DNS redirection, a client's DNS request is redirected to an authoritative DNS name server that is controlled by the CDN, which then resolves the CDN server name to the IP address of one or more content servers [11]. DNS redirection can be used to deliver full or partial site content. With the former, all DNS requests for the origin server are redirected to the CDN. With partial site content delivery, the origin site modifies certain embedded URLs so that requests for only those URLs are redirected to the CDN. The Akamai CDN uses DNS redirection to deliver partial content.

Although Akamai's network measurement, path selection and cache distribution algorithms are proprietary and private, the mechanisms that enable Akamai to redirect clients' requests are public knowledge. Below, we provide a detailed explanation of these mechanisms. The explanation is based on both publicly available sources [12]–[15] and our own measurements.

A. DNS Translation

Akamai performs DNS redirection using a hierarchy of DNS servers that translate a web client's request for content in an

Akamai customer's domain into the IP address of a nearby Akamai server (or edge server). At a high level, the DNS translation is performed as follows. First, the end user (*e.g.*, a web browser) requests a domain name translation to fetch content from an Akamai customer. The customer's DNS server uses a canonical name (CNAME) entry containing a domain name in the Akamai network. A CNAME entry serves as an alias, enabling a DNS server to redirect lookups to a new domain. Next, a hierarchy of Akamai DNS servers responds to the DNS name-translation request, using the local DNS server's IP address (if the client issues DNS requests to its local DNS) or end user's IP address (if the DNS request is issued directly), the name of the Akamai customer and the name of the requested content as a guide to determine the best two Akamai edge servers to return.

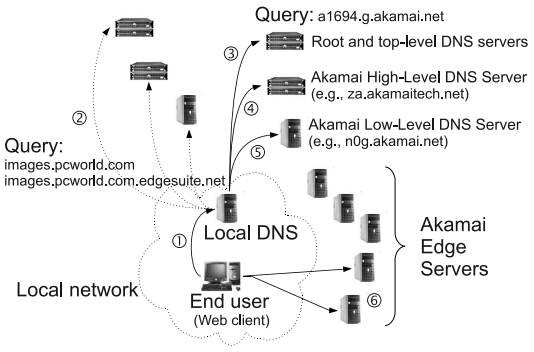


Fig. 1. Illustration of Akamai DNS translation.

Figure 1 provides a detailed example of an Akamai DNS translation, which is explained in depth in [1]. In summary, the Akamai infrastructure returns the IP addresses of *two* Akamai edge servers that it expects to offer high performance to the web client. Finally, the IP address of the edge server is returned to the web client, which is unaware of any redirection.

B. System Dynamics

It is important to note that many of the steps shown in Figure 1 are normally bypassed thanks to local DNS server (LDNS) caching. Unfortunately, this same caching can reduce a CDN's ability to direct clients to optimal servers. To ensure that clients are updated on the appropriate server to use, Akamai's DNS servers set relatively small timeout values (TTL) for their entries. For example, the TTL value for an edge server's DNS entry is 20 seconds. This means that the LDNS should request a new translation from a low-level Akamai DNS server every 20 seconds. While nothing requires an LDNS to expire entries according to their given timeout values [16], we will show how this behavior does not impact the results of our work since we request DNS translation *directly*.

III. Measuring Akamai

In this section, we present details of our large-scale measurements of the Akamai CDN. These measurements reveal important system parameters, such as the scale and dynamics of Akamai-driven redirections, which we exploit later in the paper. In particular, we answer the following questions: (i) What is the server diversity, *i.e.*, how many Akamai edge servers does an arbitrary web client "see" over long time intervals? (ii) What is the impact of clients' locations on server diversity? (iii) How does Akamai's content (*e.g.*, Yahoo vs. The New York Times) impact server diversity? (iv) What is the redirection frequency, *i.e.*, how often are clients directed to a different set of edge servers?

For our measurements we relied on 140 PlanetLab (PL) nodes scattered around the world [17]. We deployed measurement programs on 50 PL nodes in the US and Canada, 35 in Europe, 18 in Asia, 8 in South America, 4 in Australia, and the other 25 were randomly selected among the remaining PL nodes. Every 20 seconds, each of the 140 nodes independently sends a DNS request for one of the Akamai customers (*e.g.*, `images.pcworld.com`), and records the IP addresses of the edge servers returned by Akamai. The measurement results are then recorded in a database for further processing and analysis. The following results are derived from an experiment that ran continuously for 7 days. We measured 15 Akamai customers, including the following popular ones:

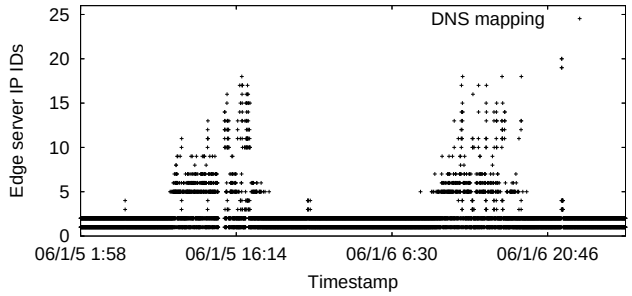
`Yahoo`, `CNN`, `Amazon`, `AOL`, `The New York Times`, `Apple`, `Monster`, `FOX News`, `MSN`, and `PCWorld`.

A. Server Diversity

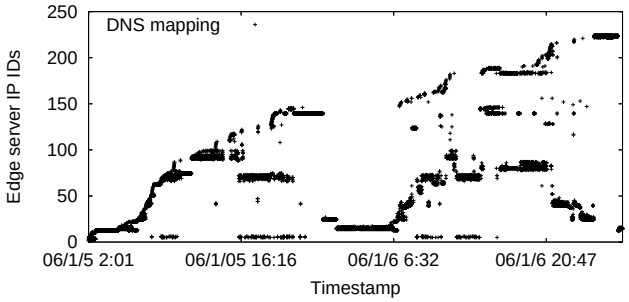
We first explore the number of unique Akamai edge servers that an arbitrary endpoint sees over long time scales. Such measurements reveal important relationships between clients and servers: A moderate number of servers seen by a client (*e.g.*, 2) reveals close proximity between the client and servers. On the other hand, clients that are farther away from the Akamai network can see a large number (*e.g.*, hundreds) of distinct Akamai servers over longer time scales. In either case, by pointing to the best servers over shorter time scales, the Akamai CDN reveals valuable path-quality information, as we demonstrate in Section IV.

Figure 2 plots the *unique* Akamai edge-server IP identification numbers (IDs) seen by two clients requesting `a943.x.a.yimg.com`, which is a CNAME for `Yahoo`. The clients are hosted on the `berkeley.intel-research.net` and `cs.purdue.edu` networks, and the result is shown over a period of two days. We plot the Akamai server IDs on the y-axis in the order of appearance, *i.e.*, those showing up earlier have lower IDs. As indicated in the figure, low-level Akamai DNS servers always return the IP addresses of *two* edge servers for redundancy, as explained in the previous section. Thus, there are always at least two points in Figure 2 corresponding to each timestamp on the x-axis.

In addition to revealing the targeted number of unique Akamai server IDs, Figure 2 extracts valuable dynamic information. Indeed, both figures show strong time-of-day effects. During the evening, both clients are directed to a small set of edge servers; during the day, they are redirected to a significantly larger number of servers. In the next section, we demonstrate that these redirections are driven by network conditions on the paths between clients and edge servers,



(a) From Berkeley



(b) From Purdue

Fig. 2. Server diversity from two characteristic PL nodes.

which change more dramatically during the day. In general, the time-of-day effects are stronger in scenarios where both a client and its associated Akamai edge servers reside in the same time zone (*e.g.*, the Berkeley case). As the edge servers are drawn from a larger pool, they tend to be scattered across a larger number of time zones (*e.g.*, the Purdue case) and the effect becomes less pronounced.

A key insight from Figure 2 is the large discrepancy between the number of unique Akamai edge servers seen by the two hosts. The Berkeley node is served by fewer than 20 unique edge servers during the day, indicating that this node and its Akamai servers are nearby. On the other hand, the lack of Akamai caching servers near the Purdue PL node significantly impacts the number of servers seen by that node — more than 200 in under two days. The majority of the servers are from the Midwest or the East Coast (*e.g.*, Boston, Cambridge, Columbus, or Dover); however, when the paths from Purdue to these servers become congested, redirections to the West Coast (*e.g.*, San Francisco or Seattle) are not unusual.

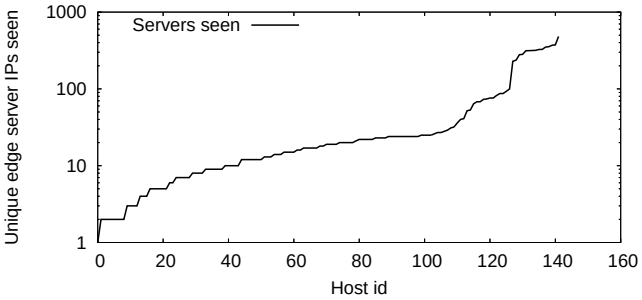


Fig. 3. Server diversity for all measure PL nodes.

Figure 3 summarizes the number of unique Akamai edge servers seen by all PL nodes from our experiments requesting the same CNAME for `Yahoo`. The number ranges from two (*e.g.*, `lbnl.nodes.planet-lab.org`), to up to 340, which is the number of servers seen by `att.nodes.planet-lab.org`. As discussed above, PL nodes experiencing a low server diversity typically share the network with Akamai edge servers.

Table I depicts the relationship between the number of edge servers seen by a PL node (requesting the same CNAME for `Yahoo`) and the average RTT to those servers. We cluster edge servers in the same class C subnet, based on our observation that Akamai edge servers that are co-located in the same data center are in the same subnet and exhibit essentially identical network characteristics (e.g., RTT to their clients). Each row lists the average number of edge server clusters seen by a PL node within a particular RTT range. For instance, when PL nodes are on average less than 5 ms away from their edge servers, they see a small number of edge server clusters (2.18 on average).

RTT (ms)	Avg. number of clusters
$[0, 5]$	2.18
$(5, 50]$	4.58
$(50, 150]$	13.77
> 150	45.25

TABLE I

AVERAGE NUMBER OF CLUSTERS SEEN BY PL NODES WHEN AVERAGE RTT LATENCIES TO THOSE CLUSTERS FALL IN THE GIVEN RANGES.

From the perspective of an overlay network aiming to "draft behind" Akamai, such PL nodes would be good candidates for *mapping* to the corresponding Akamai servers, as we demonstrate later in Section VI. Other nodes show either moderate or large server diversity.

B. The Impact of Akamai Customers on Server Diversity

In the Akamai CDN, different edge servers may host content for different customers [15]. Such an arrangement alleviates the load placed on the servers, thus improving the speed of content delivery; at the same time, this approach provides a reasonable degree of server redundancy, which is necessary for resilience to server failures. Here, we explore how this technique impacts the PL nodes' server diversity. In essence, we repeat the above experiment, but query multiple Akamai customers in addition to `Yahoo`.

Figure 4 depicts the server diversity for a set of five PL nodes and ten Akamai customers. For the reasons explained above, both `Purdue` and `Columbia` PL nodes show a large server diversity. While the actual number of observed servers certainly depends on the Akamai customer, the cardinality is generally always high for these two nodes. The exception is FEMA's (Federal Emergency Management Agency) web site, the content of which is modestly distributed on the Akamai network; we found only 43 out of over 15,000 Akamai edge servers [2] that host this web site.

Despite the fact that some of our PL nodes are placed on the same networks as Akamai edge servers, *all* PL nodes show a

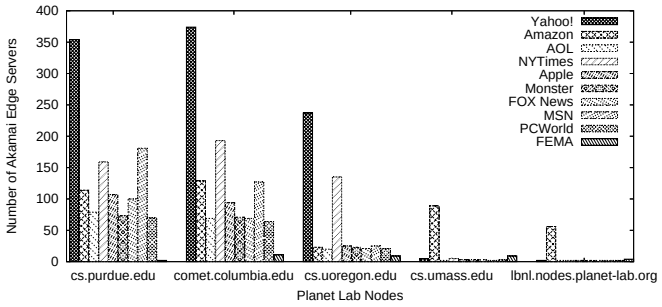


Fig. 4. Server diversity for multiple Akamai customers.

large server diversity for at least one of the Akamai customers. For example, Figure 4 indicates that querying `Yahoo` or `The New York Times` from the U. of Oregon reveals a large number of Akamai servers; likewise, querying `Amazon` from the `UMass` or `LBNL` PL nodes shows the same result. The bottom line is that because Akamai customers are hosted on different (possibly distinct) sets of servers, *all* clients, no matter how close they are to an Akamai edge server, can see a large number of servers. As we demonstrate in Section IV, a large number of servers enables clients to reveal low-latency Internet paths.

C. Redirection Dynamics

To ensure that clients are updated on the appropriate server to use, Akamai's low-level DNS servers set small, 20-second timeouts for their entries. However, nothing requires a low-level Akamai DNS server to direct clients to a new set of edge servers after each timeout. Here, we measure the frequency with which low-level Akamai DNS servers actually change their entries. In the following experiments, the PL nodes query their low-level Akamai DNS servers by requesting `a943.x.a.yimg.com` (the CNAME for `Yahoo`) every 20 seconds. By comparing the subsequent responses from the DNS servers, we are able to detect when a DNS entry is updated and measure the inter-redirection times. Our primary goal is to verify that the updates are happening at sufficiently short time scales to capture changes in network conditions.

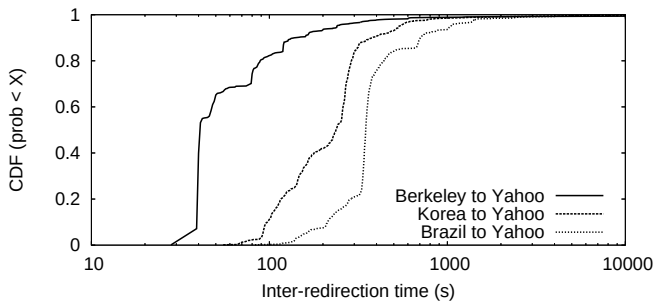


Fig. 5. Redirection dynamics from three representative nodes.

Figure 5 plots the cumulative distribution function (CDF), $F(x) = Pr[X \leq x]$, of inter-server redirection times for three PL nodes, located in Berkeley (the same node as in Figure 2(a)), South Korea, and Brazil. The CDF curve for the Berkeley node represents the inter-redirection dynamics for the vast majority of nodes in our PL set. Approximately 50% of the redirections are shorter than 40 seconds, while more than 80% of the redirections are shorter than 100 seconds. Nevertheless, very long inter-redirection times also occur, the majority of which are due to the time-of-day effects explained above.

Not all PL nodes from our set show the above characteristics. Examples are `kaist.ac.kr` and `pop-ce.rnr.br`, which are also included in Figure 5. The median redirection time is around 4 minutes for the former, and as much as 6 minutes for the latter. Moreover, the steep changes in the CDF curves reveal the most probable (still quite long) redirection time scales. As we demonstrate below, longer redirection intervals can prevent corresponding clients from achieving desirable performance if network conditions change during that period. Still, the summary statistics for the entire set of 140 PL nodes reveals satisfactory redirection intervals: the median redirection time is below 100 seconds.

IV. Does Akamai Reveal Quality Internet Paths?

Here, we answer one of the key questions relevant to our study: Do frequent Akamai redirections correlate with *network* conditions over the paths between a client and its servers? In an earlier study, Johnson *et al.* [18] demonstrated that Akamai generally picks servers that yield low client-perceived latencies. However, both network- and server-side effects impact the overall latency, and Akamai claims to perform and use both measurements to redirect clients to the closest server [2]. *decouple* the network side from the server side to determine which one dominates performance. If the server component prevails, then only Akamai's clients benefit from redirections. However, if the network component dominates, then redirections reveal network conditions on Internet paths – information valuable to the broader community of Internet users.

A. Methodology

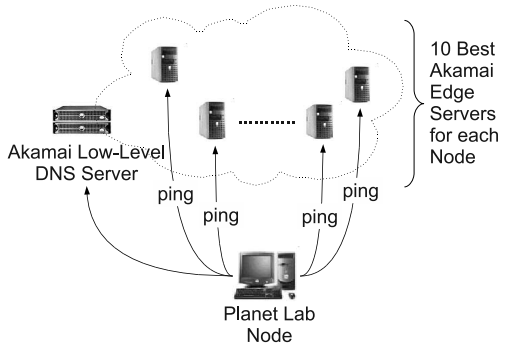


Fig. 6. Illustration of Measurement Methodology.

Figure 6 illustrates our measurement methodology for determining whether Akamai redirections reveal quality Internet paths. As in the above experiments, each of the 140 nodes periodically sends a DNS request for one of the Akamai

customers and records the IP addresses of the edge servers returned by Akamai. To capture *every* redirection change affecting our deployment, we set the DNS lookup period to 20 seconds, the same TTL value set by Akamai's low level DNS servers (as discussed in Section II). In order to determine the quality of the Internet paths between PL nodes and their corresponding Akamai servers, we perform ping measurements to Akamai edge servers during each 20-second period. In particular, every 5 seconds, each PL node *pings* a set of the 10 best Akamai edge servers. That is, whenever a new server ID is returned by Akamai, it replaces the longest-RTT edge server in the current set. In this section, we use the average of the four ping measurements to an edge server as the estimated RTT. It is essential to understand that by pinging, instead of fetching parts of Akamai-hosted pages from servers as done in [18], we effectively avoid measuring *combined* network and server latencies, and isolate the network-side effects. Finally, the results of 7 days of measurements from all 140 nodes are collected in a database and processed.

B. Normalized Rank

The latency between a client and its servers varies depending on the client's location. For example, the latencies for nodes located in the middle of Akamai "hot-spots" are on the order of a few milliseconds; on the other hand, the RTTs of other nodes (*e.g.*, located in South America) to the closest Akamai server are on the order of several *hundreds* of milliseconds. To determine the relative quality of paths to edge servers selected by Akamai, we introduce the *rank* metric. Rank represents the correlation of Akamai's redirection decisions to network latencies. In each 20-second-long round of the experiment, the 10 best Akamai paths are ranked by the average RTTs measured from the client, in the order from the longest (0) to the shortest (9). Since Akamai returns IP addresses of two edge servers in each round, we assign ranks r_1 and r_2 to the corresponding edge servers. We define the total rank, r, as $r = r_1 + r_2 - 1$. If the paths returned by Akamai are the best two among all ten paths, the rank is 16; similarly, if the Akamai paths are the worst in the group, the rank equals zero. Finally, the *normalized rank* is simply the the rank multiplied by $100/Maximum\_Rank$, where $Maximum\_Rank$ is 16.

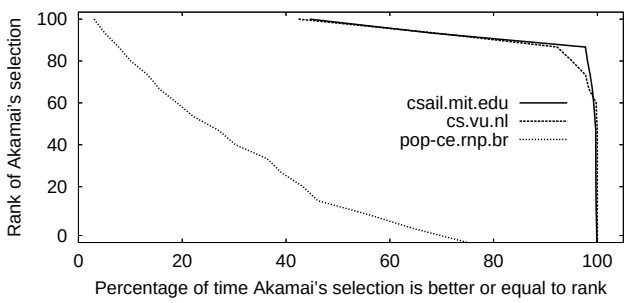


Fig. 7. Normalized ranks for three characteristic PL nodes.

Figure 7 plots the normalized rank of Internet paths measured from the sources indicated in the figure to the Akamai servers. A point in the figure with coordinates (x,y) means that the rank of the two paths returned by Akamai is better than or equal to the rank y during x percent of the duration of the 7-day experiment. Thus, the closer the curve is to the upper right corner, the better the corresponding paths selected by Akamai. Indeed, Figure 7 indicates that the Akamai redirections for `csail.mit.edu` and `cs.vu.nl` almost perfectly follow network conditions. On the other hand, because the average redirection interval is quite high in the Brazil case (6 minutes, as shown in Figure 5), we observe a relatively poor selection of servers in terms of path latency. Indeed, even a random or round-robin path selection over shorter time intervals would achieve a better result in this case.

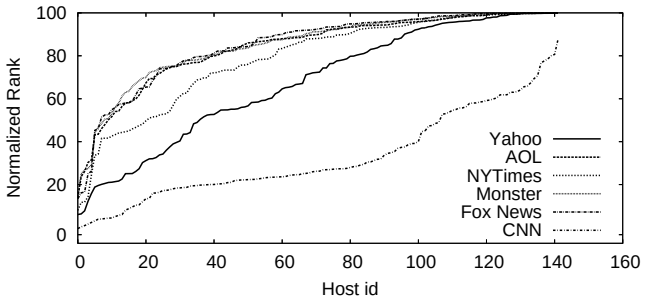
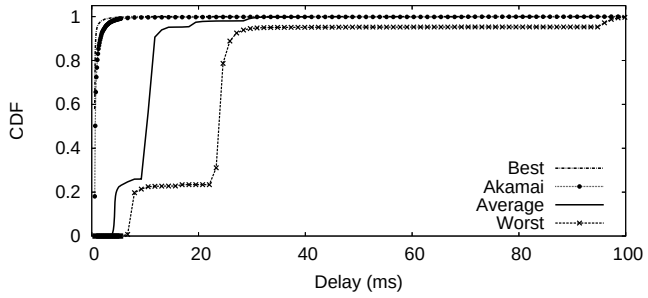


Fig. 8. Normalized rank for all PL nodes.

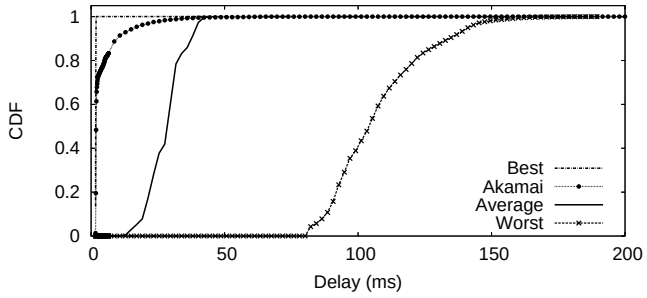
Figure 8 depicts the *normalized* rank of the paths that Akamai returns to all 140 PL nodes. We plot the curves in increasing rank order. Hence, the order of PL node IDs on the *x-axis*, while similar, is *not* identical for different customers. To examine the effect of Akamai customer on this metric, we measure six different Akamai customers as indicated in the figure. The key insight from the figure is that Akamai redirections strongly correlate with network conditions for most of its customers.

For example, for the set of customers showing the best performance (*e.g.* `Fox News`), more than 97% of the paths chosen by Akamai (IDs 5-140 in Figure 8) are better than average (normalized rank larger than 50); at the same time, more than 70% of Akamai paths are approximately among the best 10% of paths (normalized rank larger than 90).

Figure 8 also reveals that Akamai offers different performance depending on the customer. As we previously explained, this occurs because different Akamai servers can, and often do, host a different set of customers. It is interesting, however, to find that `CNN` (CNAME `a1921.aol.akamai.net`) shows by far the worst result in our measurement. Further investigation showed that all of the edge servers we found for CNN are from the same region in the US; moreover, they are all from the same subnet. This finding seems to contradict Akamai's policy of using globally deployed edge servers to serve content. We later learned that none of CNN's servers are currently owned by Akamai. Therefore, it appears that CNN is no longer an Akamai customer, though they still have "akamai.net" as the



(a) MIT



(b) Amsterdam

Fig. 9. CDF of RTTs for Akamai paths for two PL nodes.

postfix of their CNAME. [1] For this reason, we removed CNN from all other figures in this study. Regardless, this finding provides evidence that CDN services that utilize network measurements and global server deployment are significantly better than traditional web content distribution using load-balancing server farms in a few data centers.

C. Latency

In this subsection, we measure the *latency* gains made possible by following the paths to edge servers returned by Akamai. Such measurements not only reveal the performance of the Akamai CDN, but also indicate the spectrum of potential latency gains achievable with Akamai-driven one-hop source routing, which we explore in the next section.

For each PL node, we collect the statistics for the RTTs on the paths between the client and the Akamai servers as follows. (i) Best delay, defined as the lowest RTT in *each 20-second-long measurement round* among the current ten best Akamai paths. (ii) Akamai's delay, defined as the *average* of RTTs on the paths to the two edge servers *selected* by Akamai in each measurement round. (iii) Average delay, defined as the average of the ten best Akamai-recommended paths. (iv) Worst delay, defined as the highest RTT in each measurement round among all ten paths.

Figure 9 plots the CDF curves for two PL nodes, `csmail.mit.edu` and `cs.vu.nl`, previously shown in Figure 7. Both figures confirm that Akamai indeed does a great job in these two scenarios; the Akamai path is almost identical to the best path in both cases. However, the key insight from the figure is that the relative latency gain depends on the distance between the PL node and its Akamai edge servers. For example, the `MIT` (Cambridge, MA) node obviously operates in an Akamai hot-spot, since the difference between the medians (CDF = 0.5) of the best and the worst path is only 20 ms. On the contrary, the corresponding difference is as much as 100 ms in the Vrije U. (Amsterdam) case. Indeed, as the distance between the Akamai CDN and a PL node increases, both the number of servers (and paths) increases and the variance of path quality increases. Thus, following the Akamai redirections brings the largest latency gains for nodes that are distant from their edge servers.

[1] This has changed since our original evaluation and CNN is currently supported by a different CDN. CNN no longer uses the CNAME shown in this paper.

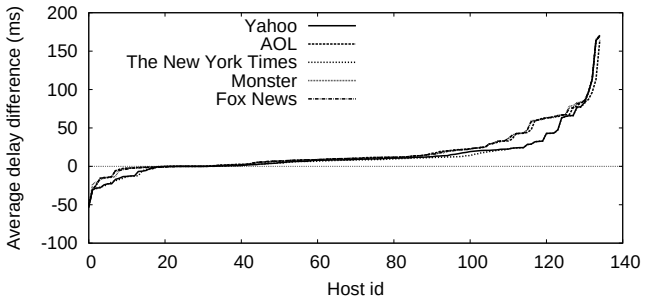


Fig. 10. Latency gains for all measured PL nodes.

Figure 10 plots the latency performance over all 140 PL nodes for different Akamai customers. For each node, we compute the average difference between (i) the RTT corresponding to the average of the ten best Akamai paths seen by the node, and (ii) the RTT corresponding to the Akamai path. The *y-axis* of Figure 10 plots the average difference between the two paths. Thus, nodes on the left of the figure (*e.g.*, 0-20) show the worst performance, *i.e.*, the path that Akamai selects is worse than the average of the current ten best Akamai paths. We found that this group is dominated by nodes that have a large server (and path) diversity and a small redirection frequency. Nodes with IDs in the range 20-30 are dominated by a small number of short-latency Akamai paths; in this case, although Akamai redirections correlate well with measured network latencies, the difference among the paths is negligible, *e.g.*, less than 1 ms. Finally, the vast majority of nodes (IDs 30-140) are directed to better-than-average paths. For a large percentage of nodes (IDs 50-140), the gains are quite substantial, ranging from 10 ms to 170 ms.

To summarize, we demonstrated that Akamai redirections overwhelmingly reveal the *network* conditions over the paths between end-users and Akamai edge-servers. Thus, by querying low-level Akamai DNS servers, an endpoint can reveal information about quality Internet paths *without* extensively probing and monitoring them. While this is potentially useful for many applications, in the next section, we necessarily focus on one such application.

V. Akamai-driven One-hop Source Routing

In this section, we examine the potential for performance improvement by using Akamai to drive an example network

application: one-hop routing in a large-scale overlay network. Since Akamai redirections generally reveal low-latency paths between nodes and edge servers, such an overlay network can use these redirections to route its own traffic. Even if an application is not primarily interested in low latency, but rather strives for high-bandwidth, the Akamai-driven approach is still viable. Measurements from [19] indicate that the vast majority of TCP flows are limited by the receiver advertised window parameter. Similarly, a recent measurement study shows that as much as 85% of the KaZaA clients do not suffer any packet loss [20]. Hence, in such cases, lower latencies directly translate to larger throughputs [21].

The key prerequisite in this environment is for the overlay network to be able to map a subset of its nodes to Akamai edge servers. Fortunately, the number of nodes in large-scale peer-to-peer (P2P) networks (*e.g.*, KaZaA [22]) is typically several orders of magnitude larger than the total number of Akamai servers; thus, finding hosts that share networks with Akamai edge servers should not be difficult, as we demonstrate in the next section. Moreover, Akamai deploys its edge servers within ISPs' networks at no charge [10]. This both greatly reduces ISPs' bandwidth expenses and improves the performance of Akamai's clients; likewise, it increases the number of potential overlay nodes that can map their positions to Akamai servers.

As a concrete example of how Akamai-driven, one-hop source routing works, consider two nodes in a large-scale overlay network. To find a high-quality path between them, the nodes perform a "race" to determine which path has the smallest latency: the direct path between the two nodes, or the one-hop path via a third node mapped to an Akamai server. In our scenario, the Akamai path consists of two parts. The first is the path from the source node to the (frequently updated) Akamai edge server; the second part is the path from the Akamai edge server to the destination. As we showed above, by selecting low-latency Internet paths, Akamai manages to successfully avoid network hot spots; this can potentially improve the end-to-end (source — Akamai node — destination) path performance.

Of course, the Akamai path is not always better than the direct path. For example, consider two nodes in Brazil, a country poorly served by Akamai. In this case, the nodes should clearly use the direct path since the Akamai servers are likely to be located in the US. Despite the potential for performance degradation, we will show that it is possible to identify and use the better of the two types of paths – without a large measurement overhead.

A. Methodology

Figure 11 depicts the experimental setup for the measurements in this section. For each pair of nodes, one node is designated as the source and the other as the destination. Throughout the experiment, we measure the RTTs for 11 paths between the source and the destination. The first path is the direct one, which the source node measures by pinging the destination and recording the corresponding RTT. The other ten paths are "recommended" by Akamai, and we measure

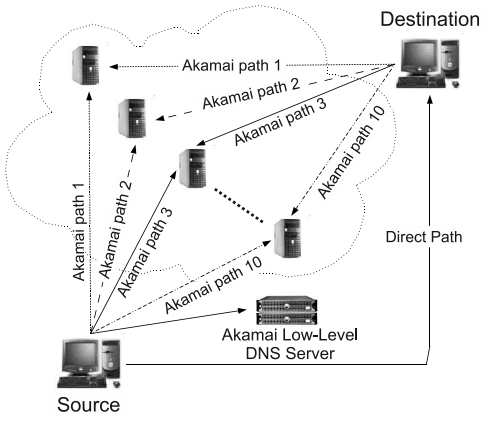


Fig. 11. Illustration of the measurement methodology.

their RTTs as follows. The source node iteratively issues a DNS query for an Akamai customer. In Figure 11, it repeatedly requests `a943.x.a.yimg.com`, which is the CNAME for `Yahoo`. As in the previous experiment, the source node measures and records the RTTs to the 10 current, lowest-latency Akamai edge servers it witnessed. Additionally, the source node notifies the destination node of the IP addresses for those 10 Akamai edge servers. This enables the destination node to measure RTTs to the most recent edge servers that the source node has witnessed. Finally, by adding the corresponding RTTs measured from the source and the destination to the set of Akamai servers, we estimate the RTTs of the 10 distinct one-hop paths.

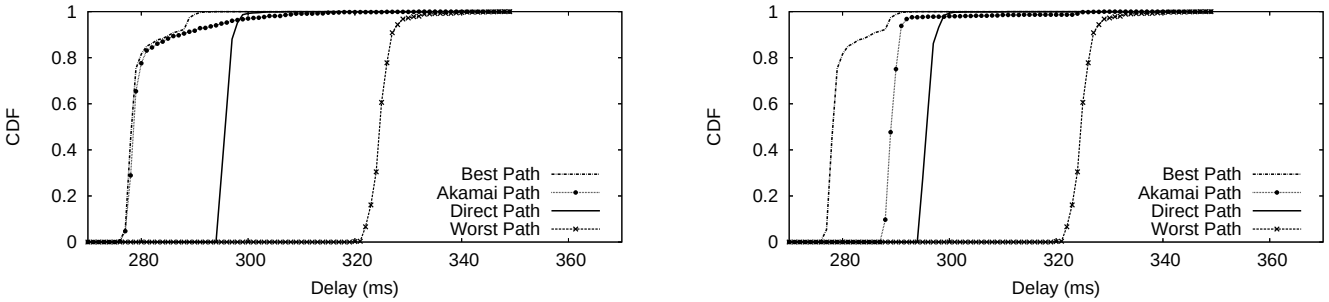
Note, however, that by evaluating the path through an Akamai edge server we are not accounting for the potential effects of detouring through overlay nodes mapped to those Akamai edge server. As such, the presented results should be understood as an upper bound on the actual performance. In Section VI, we discuss and evaluate the actual performance of routing via intermediate overlay nodes.

Another important characteristic of the above measurement is its asymmetry. For the same pair of nodes, the results can be quite different depending on which node is assigned to be the source. This occurs because the Akamai servers witnessed by the source and destination nodes are generally different, particularly for geographically distant nodes. We explore such effects in more detail below.

B. A Case Study: Taiwan — UK

To demonstrate the potential of Akamai-driven one-hop source routing, and to show the effects of asymmetry, we initially present results for a pair of geographically distant PL nodes. The first is `iis.sinica.edu.tw`, located in Taiwan, and the second is `cambridge.intel-research.net`, located in the UK.

Figure 12 plots the CDF functions of the RTTs for the following paths: (i) Best path, defined as the path with the lowest RTT in *each 20-second-long measurement round* among the ten one-hop paths and the direct path; (ii) Akamai's path, defined as the *average* of the two one-hop paths via the



(a) Taiwan — UK (b) UK — Taiwan

Fig. 12. CDFs for path latencies between Taiwan and the UK. In each figure, the first country is designated as the source.

two edge servers selected (and frequently updated) by Akamai; (iii) Direct path, measured from the source to the destination; and (iv) Worst path, defined as the path with the highest RTT in each measurement round among all eleven (ten one-hop and one direct) paths. In Figure 12(a), the Taiwan node is the source; in Figure 12(b), the UK node is the source.

The CDF curves from Figure 12 illustrate the gains that can be achieved by using Akamai's one-hop paths. For example, Akamai's path is nearly optimal in the Taiwan case, outpacing the direct path by nearly 20 ms. On the other hand, while the paths chosen using Akamai's "recommendations" from the UK are suboptimal, they still generally beat the direct path. To shed more light on these results, we collected statistics for the Akamai servers seen by each node. For the Taiwan node, 80% of edge server "hits" are in Tawain, 15% in Japan, and 5% in the US. For the UK's node, 75% of the hits are in the UK, and 25% are in the US. The large number of servers close to (in the same country as) the source nodes indicates that the gains over the direct path come from avoiding hot spots close to the sources. Moreover, whenever the "middle nodes" are not in the country of origin, they are still placed along a high quality path to the destination, thus improving the performance.

C. Aggregate Results

In this section, we study a much broader range of sources and destinations to determine the performance of Akamai-driven one-hop source routing. For this study, we assemble a list of PL nodes located in the US (6), Europe (3), Asia (3), and South America (2). We then pair all nodes with each other by randomly choosing the source and destination for each pair. Out of 91 attempted paths, 78 successfully completed 3-day-long experiments, while the rest failed (*e.g.*, due to a PL node rebooting).

Figure 13 illustrates the difference between the latency using a direct path and the Akamai one-hop paths, all measured over short, 20-second-long time scales. For each pair, we compute the best and the worst (out of ten) one-hop Akamai paths, and the average of the two one-hop paths returned by Akamai. A negative value means that the corresponding direct path is better for the pair; otherwise, the Akamai-driven path is better. For example, the best, worst, and Akamai-selected paths are, on average, worse than the direct path for Pair ID 1. On the other hand, all one-hop Akamai paths outperform the direct path for pair ID 78.

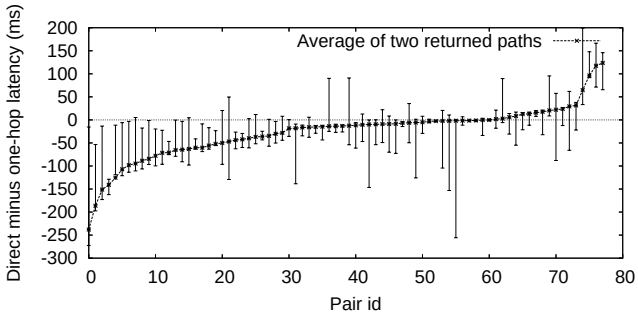


Fig. 13. Latency differences between one-hop routing and direct-path routing.

The figure indicates that in approximately 25% of scenarios (IDs 60-78), Akamai-driven paths outperform the direct path, in the same manner as discussed for the Taiwan—UK example above. The majority of the paths are intercontinental, excluding South America, *e.g.*, Asia—US, US—Europe, Europe—Asia. The second group (path IDs 30-60) is dominated by the intra-Europe or intra-US paths for which the potential gains of detouring are smaller in general. Finally, the third group (path IDs 0-30) consists of two subgroups: (i) PL nodes that are close to each other and relatively far away from Akamai servers (*e.g.*, Korea—Japan) that have a better direct path and (ii) all paths sourced in South America see no gain from Akamai due to infrequent refreshing of the low-level Akamai DNS server's tables.

Finally, an important point is that in approximately 50% of scenarios, the best measured Akamai one-hop path outperforms the direct path. This indicates both that detouring has a significant potential for improving performance, and that Akamai is successful in *locating* quality detouring points. These results encouraged us to investigate to what extent we can capture these performance improvements using a low-cost, deployable algorithm.

D. Path Pruning

We now discuss practical techniques for determining which of the following two paths to use for routing: the direct or the Akamai-recommended path. Thus, while the measurement overhead is already significantly reduced (to the above two paths), the question is whether such comparisons can be directly avoided by pruning low-quality paths from the set of available ones. Overall, there are two issues to consider:

(i) the best frequency at which one should reconsider whether to use a one-hop route or simply opt for direct one and (ii) if opting for detouring, whether to use the first edge server returned or to measure the alternative ones to determine the better one.

To determine the effectiveness of using Akamai-driven one-hop routing, we must develop a low-cost algorithm that (i) enables nodes to reap any performance benefits of Akamai's network measurements and (ii) quickly determines when Akamai cannot improve performance and one should stick to the direct path between the nodes. In short, we want to find the best path among the direct path and Akamai-recommended paths, while minimizing the number of network measurements required to find that path. Thus, the algorithm must find a good trade-off between resulting network performance and measurement overhead.

We evaluate four heuristics for determining routes. First, we consider how frequently the algorithm should reevaluate the decision to use the direct path or one-hop paths. This decision can be made (i) once for the duration of the experiment (static) or (ii) it can be reevaluated every y minutes (dynamic). In either case, if a one-hop path is selected, we explore the performance of two alternatives:

- *First Akamai Server (FAS)*. We query the Akamai DNS for an edge server approximately once per minute and use the first server returned by Akamai as our one hop.
- *Better of the Two Akamai Servers (BTAS)*. We query the Akamai DNS for an edge server approximately once per minute. We perform ping measurements to compare the quality of the paths along the two edge servers returned by DNS and use the lower-latency path.

For the static algorithms, we must include a bootstrap period that enables the Akamai network to "warm up," *i.e.*, to determine which servers are best for our measurement nodes. For the following experiments, we use a bootstrap period of approximately 100 minutes.

To form a baseline for comparing the effectiveness of these algorithms, we first determine the maximum latency gain that can be achieved by pruning paths. For example, if the *best* (out of 10) one hop Akamai path is 100 ms *faster* than the direct path, the maximum gain is 100 ms. Similarly, if the *worst* (out of 10) Akamai path is 100 ms *slower* than the direct path, then the maximum gain is again 100 ms. We aggregated the maximum latency gain over all 78 pairs of nodes and used the average value as our baseline. Figure 14 depicts the performance of our four algorithms relative to the maximum latency gain (100%) and to the case where the direct path is always used (*i.e.*, no routing decisions are made).

Figure 14 shows that using the direct path alone accounts for only about 78% of the performance gain seen in our experiments. This further shows that Akamai is good at locating nodes along high-quality paths. The figure also clearly shows that the dynamic versions of FAS and BTAS can lead to significant improvement over the direct path. In particular, the update frequency for BTAS and FAS can be as long as almost 2 hours before its performance significantly declines. Even with update intervals on the order of a day, these algorithms outperform the direct path on average. It is also clear that

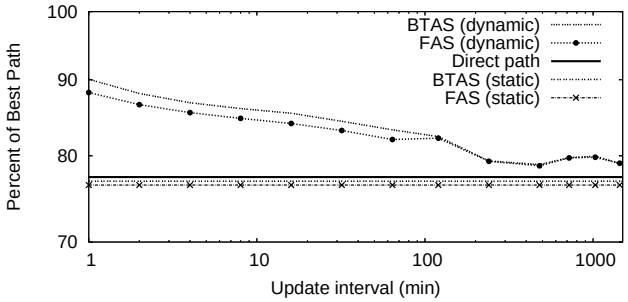


Fig. 14. Comparing pruning algorithms against best-case performance.

BTAS outperforms FAS over shorter timer intervals (on the order of hours). As expected, choosing the better of the two Akamai paths is more resilient to changes in path quality than when simply picking the first of the two. Still, the difference is not dramatic.

We also note that the performance of the static versions of the pruning algorithms are nearly identical, and are slightly worse than using the direct path (by $\leq 1\%$) when averaged over all pairs of nodes. As discussed above, Akamai optimizes only the source part of the one-hop path, and thus may sometimes direct clients to edges that are along slower one-hop paths than the direct path. Since the static versions of these algorithms cannot "double-check" Akamai, they may suffer a performance hit by sticking with their original decision to always use one-hop paths.

VI. Detouring Through CDN-Associated Nodes

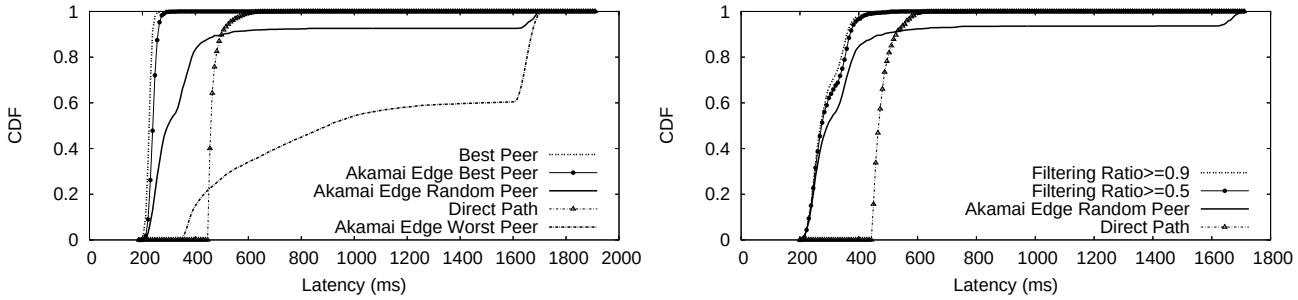
To better understand the potential benefits of Akamai-driven, one-hop detouring in scenarios with tens and hundreds of *thousands* of nodes – well beyond what is feasible with today's PlanetLab – we collected IP addresses for a large number of hosts in the BitTorrent network and used them as potential peers in a large overlay-based system. Based on our measurement results, we provide answers to the following question: How do peer-to-CDN mappings affect the quality of CDN-driven one-hop source routing?

A. Measurement Methodology

Over a three-month long measurement on the BitTorrent network, we connected to multiple torrents for free software (including OpenOffice.org releases and several Linux distributions such as SuSE and Debian), and were able to gather a large number of unique BitTorrent peer IP addresses. We obtain and record CDN redirections experienced by the observed peers by performing *remote DNS lookups* on these peers' behalf. [2]

From the set of collected BitTorrent IP addresses, we randomly select 10,000 located in networks with DNS name servers that respond to recursive queries. Using an approach that resembles that used by Gummadi et al. [23], we employ these servers to gather CDN redirections dynamics using *recursive DNS queries*.

[2]Using the `dig(1)` utility.



(a) One-hop paths vs. direct path

(b) One-hop path filtering

Fig. 15. Akamai-driven Detouring Through CDN-Associated Peers

We recorded 6,523 unique Akamai replica server IP addresses using recursive measurements on behalf of the 10,000 BitTorrent peers. It is common for a CDN to place a number of servers in each data center, for both load balancing and failure tolerance [12], [14]. Thus, we cluster the observed IP addresses based on their class-C subnet IP prefix[3] to form 618 CDN clusters. In this manner, we transform the mapping problem such that it is between peers in the wide-area system and CDN replica server clusters. Finally, we filter out statistically insignificant clusters (i.e., that are rarely seen by BitTorrent peers – with a hit ratio below 0.001%), leaving us with a working set of 305 CDN clusters.

B. Mapping-based Path Quality

We have shown that in approximately 50% of scenarios, the Akamai-recommended one-hop path outperforms the direct path. Naturally, we focus on such scenarios here. Our previous results, however, employ edge servers as intermediate nodes for detouring. Since we cannot use CDN's edge servers for detouring our traffic, we must instead locate overlay nodes that are close to those edge servers. In this section, we demonstrate that one-hop source routes through *peers* associated with CDN edges do retain gains characteristic for one-hop source routes through these edges.

We select 22 Internet paths for experimentation, the majority of which are inter-continental. As shown in section V-C, longer paths benefit the most from CDN-driven detouring. For each of the 22 source-destination pairs, we identify a set of peers that can be mapped to the Akamai-recommended edge servers. Then, for each source-destination pair, we measure RTTs for $n + k + 1$ paths between the source and the destination in each one-minute-long round. The first path is the direct one, which the source node measures by pinging the destination and recording the corresponding RTT. The k additional paths are single-hop routes measured through recommended Akamai edge servers. The final n paths are single-hop routes measured through BitTorrent peers that are *mapped* to the above k edges. The following paragraphs focus on a representative path between `(cs-ipv6.lancs.ac.uk)` and `(iiitb.ac.in)`.

Figure 15(a) plots the CDF functions of the RTTs for the direct path between these two hosts and a number of "alternative" detouring paths. The curve marked by "Best Peer" denotes the statistics for the lowest-latency *path* among all measured one-hop paths through peers in each one-minute round. It represents the performance upper bound. The "Akamai Edge Best Peer" curve shows the RTT statistics for a path that strictly follows Akamai redirections, (*i.e.*, potentially changes each minute); still, it is routed through the shortest-RTT peer associated with each edge. "Akamai Edge Random Peer" also strictly follows Akamai redirections; yet, it represents the *average* RTT of all peers associated with a particular edge server. Finally, "Akamai Edge Worst Peer" is an Akamai-driven path that is routed through the worst peer (longest-RTT) associated with each edge server.

Figure 15(a) shows that routes through *randomly-selected* peers (associated with appropriate CDN edges) outperform the direct path by approximately 150 ms. Thus, even a simple heuristic manages to retain gains common for endpoint-to-edge-server paths. However, Figure 15(a) also clearly shows that there is large variability in the quality of Akamai-driven paths that use intermediate peers. For example, routing through the best peer is close to optimal, yet, routing through the worst path could be disastrous. Also, it is evident that the worst path significantly biases the average path ("Akamai Edge Random Peer") statistics, yielding a long distribution tail (ending at 1,600 ms). We posit that this problematic paths are primarily due to weak mappings between peers and edge servers. To avoid them, we simply filter mappings based on the frequency with which a peer is redirected to a given CDN cluster. The intuition behind this filtering is that if a peer rarely finds a given edge server cluster over long periods of time, their association is most probably not due to measured network conditions. On the other hand, if a peer is frequently redirected to a particular cluster it is likely the two are indeed close and, therefore, are exposed to similar path-quality characteristics.

Figure 15(b) demonstrates that this is indeed the case. For example, with the ≥ 0.5 filtering rule (a mapping is valid if it occurs over 50% of the time), we can filter out poor mappings and eliminate the long tail. Not surprisingly, the ≥ 0.9 filtering rule further improves the performance of CDN-based detouring, however, the 50% rule is sufficient to filter the majority of poor paths. We conclude this section by noting that measurements over the rest of the investigated paths yield similar results. Further, in *all* the scenarios that

[3]A class C subnet has 256 IP addresses.

we explored, whenever Akamai-driven paths through CDN servers outperform the direct path, so does, *on average*, a randomly-selected path through the nodes filtered out by the 50% mapping rule.

VII. Discussion and Related Work

In this section, we discuss several issues relevant to our study and present work related to topics covered in this paper.

A. Discussion

Akarouting. Akamai has formed a private, proprietary network measurement system and overlay network. One goal of this system is to improve download speeds by using its network measurements to find "high-speed" Internet paths over which to transfer content *within* the network. In addition, Akamai applies one-hop source routing to transfer content from customer origin servers, *e.g.*, `NYTimes.com` to edge-servers, a technique they call Akarouting [8]. Unlike Akarouting, our approach uses Akamai's client-to-server redirections to locate and utilize potentially high-quality detouring points in a *separate* overlay network.

Additionally, it is important to note that the Akamai-driven one-hop source routes in our experiments are *not* routed through the proprietary Akamai-owned network. Our ping and trace-route measurements confirm that the routes from PL nodes to Akamai edge-servers take "public" Internet paths available to everybody.

"Free riding" on Akamai. It is very possible that Akamai, or any other CDN, might not be excited by the fact that third parties are exploiting the CDNs' measurements for their own purposes. However, it is important to realize that the load placed on the Akamai DNS infrastructure by our proposed technique, even in the case of larger-scale deployment by overlay networks, will probably be negligible compared to the load that this infrastructure is already experiencing from its "regular" clients. This is because the Akamai CDN hosts some of the most popular web enterprises and keeps small TTL values for an edge server's DNS entries. Thus, the "regular" load placed on low-level Akamai DNS servers is already very high. The implication is that the proposed technique should not jeopardize Akamai's performance. Moreover, we believe that any attempt to detect the "free-riding" nodes might face non-negligible false negatives and positives, thus unnecessarily degrading the "regular" clients' performance. Finally, it is important to understand that the proposed techniques do not require the Akamai edge servers to reply to *pings*. Although we indeed used pings to *verify* the correlation between network latencies and Akamai redirections, they are not required for conducting Akamai-driven source routing.

The implications of widespread adoption. Finally, one concern is that if the approach is successful and widely followed, then this might affect the performance of the network paths that Akamai identifies as "good." For example, if *all* clients from a particular domain use Akamai's recommendations in precisely the same way, previously uncongested paths may become congested within a short time period. However, we do not expect this to be the common case, since different overlay nodes can choose to query Akamai's DNS servers for any of the large number of hosted web sites (*e.g.*, `The New York Times` vs. `Amazon`). As explained in Section III-B, different edge servers typically host different web sites. As a result, overlay nodes from the same domain will have different "views" of the network, which naturally helps to spread the traffic load along different network paths.

B. Related Work

There have been a number of studies on the use, effectiveness and impact of content distribution networks. In an early work, Gadde *et al.* [24] analyze the effectiveness of interior web caches and CDNs based on an analytical model of caching behavior. Two recent studies confirm that CDNs reduce average download response times, but that DNS redirection techniques add noticeable overhead due to DNS latency [11], [25]. In [26] the authors examined how content distribution servers improved latency when compared to throughput from the origin servers. Johnson *et al.* [18] assessed the degree to which two different CDNs optimally redirect requests among their mirrors, and argue that these CDNs appeared to use the DNS mechanisms not necessarily to select optimal servers, but to avoid selecting bad ones. Krishnamurthy *et al.* [11] do an extensive study on the use of CDNs and propose a methodology to study client-perceived performance. Saroiu *et al.* [27] characterize four content delivery systems, including Akamai, based on traces collected at a university's borders routers. In their study Akamai appears as the smallest bandwidth consumer (0.2%), with Gnutella, Kazaa and WWW traffic consuming near 60% of the remaining bandwidth. In addition to revealing and understanding CDN behavior, our research enables clients of *other* networks to reuse the measurements made by CDNs for their own purposes.

Some previous work has addressed other issues with DNS-based control, particularly in the context of server selection in CDNs. Shaikh *et al.* [6] evaluate the impact of DNS-based server selection on DNS; the authors find that extremely low TTL values (on the order of seconds) can adversely affect latency. In a related work, Jung *et al.* [28] show that the most common values for TTL should not greatly increase DNS-related wide-area network traffic. Studies by Shaikh *et al.* [6] and Mao *et al.* [29] found that clients are often not close in network topology to the name servers they use, questioning the accuracy of server selection based on IP address. No such problem exists with our scheme because clients request DNS translation *directly*, and not via a local DNS server.

CDN-driven detouring is based on, and related to overlay routing systems that attempt to improve a client's reliability and performance. Our work is the first one to propose relying on CDNs' measurements to locate and utilize quality Internet paths without performing extensive path probing or monitoring. The Detour study [30] suggested that this could be accomplished by routing via intermediate end systems. The Resilient Overlay Network (RON) project demonstrated this to be the case in a small-scale overlay [31]. This, however, required background monitoring that is not scalable and therefore limits the approach to communication among a relatively

small set of nodes. The solution proposed in [32] relies on end-to-end probing of the overlay paths and the inference of the loss probabilities on the underlying physical path segments, which suffers from similar scalability limitations. Our CDN-driven detouring technique improves the performance of the above systems, not only by more efficiently avoiding network outages and hot spots, but also by eliminating the need to probe a number of Internet paths.

In order to limit the resource requirement for overlays, more recent studies have focused on reducing the end-to-end measurement needed to select overlay paths. In [33], the authors propose a routing underlay dedicated to topology probing. With the help of this underlay, one can use inferred AS path information to construct disjoint paths between communicating nodes. The potential problem of this method is the accuracy of AS path inference. For instance, [29] showed that AS path inference can often be much less accurate than expected. Gummadi *et al.* [34] select relay nodes by randomly choosing k overlay nodes (random-k) and selecting the one with the best performance. With a small k, there is clearly the risk that random selection, while avoiding outages, will discard a good relay node. Chen *et al.* [35] present a linear algebraic approach to monitor overlay paths efficiently. They show how to measure k linear independent paths and infer packet loss rates of all other paths. The key difference between the above approaches and Akamai-driven one-hop source routing is that the former is intended to improve system's *reliability* by avoiding network outages and/or lossy network paths, while the goal of our scheme is to improve clients' *performance* by selecting and hopping over quality (low-latency) paths as recommended by Akamai.

In another closely related work, Fei *et al.* [36] use AS-level path information inferred from `traceroute` to reduce the size of the candidate set for one-hop routing. The goal is to limit the overhead in selecting middle hops by examining only nodes along paths between the origin and destination that diverge in the AS-level path as early as possible. Although this technique is shown to provide the ability to avoid performance degradation over direct paths, the main limitation is that the authors do not propose or evaluate the cost and effectiveness of *online, dynamic* techniques for selecting middle nodes according to their heuristic. Further, this technique yields anywhere from 1 to n candidate nodes to probe, whereas our proposed technique always yields at most two. Finally, the coarse resolution of AS-level path disjointness may eliminate good candidate middle nodes that would otherwise be captured by extensive measurement from a large-scale system such as Akamai.

More generally, a number of research efforts have recently begun to address some of the challenges in supporting Clark et al.'s [37] grand vision of a knowledge plane for large-scale, self-managing distributed systems [38]–[41]. How to best incorporate CDNs' network views into some of these systems is part of our future work.

Finally, our work is inspired by tools like Sting [42], T-BIT [19], and King [23], which use existing protocols "in unanticipated ways to obtain results that were previously intractable [23]."

VIII. Conclusions

In this paper, we performed an extensive measurement study of the Akamai CDN; the goal was to determine how one can *infer* and *utilize* quality, short time-scale regarding network conditions without the cost of extensive network measurement. By concurrently measuring network paths and monitoring the frequently refreshed low-level Akamai DNS server tables, we showed that: (i) Akamai-server redirections strongly correlate with network conditions on the paths between clients and servers; more than 70% of paths chosen by Akamai are among the best 10% of the measured network paths. (ii) For a given client, the correlation level predominantly depends on the inter-redirection frequency of the corresponding low-level Akamai DNS server. (iii) Due to low redirection frequencies, clients from South America experience correlation levels that are *below* that achievable by a random or round-robin path selection. (iv) Because Akamai customers are heterogeneously hosted on the edge servers, all investigated clients see a large number of servers (paths) for at least one of the customers. (v) CDN services that utilize network measurements and global server deployment can significantly outperform traditional web content distribution that use load-balancing server farms in a few data centers.

To provide a sample application, we studied the potential for utilizing Akamai redirections to drive one-hop source routes (*i.e.*, detours) in a large-scale overlay network. By concurrently measuring and comparing Akamai-driven one-hop with direct paths between nodes scattered around the globe, we show that (i) in more than 50% of investigated scenarios, it is better to route through the nodes "discovered" by Akamai than to use direct paths. (ii) in 25% of investigated scenarios, a better-than-direct path can be utilized by always following Akamai redirections at the source. (iii) The vast majority of Akamai-driven paths between Asia and Europe belong to the above category; in addition to avoiding local hot spots, they exploit rich Akamai "proxy" infrastructure placed in between the two — *e.g.*, in the US. (iv) Other nodes can apply simple, low-overhead techniques to decide whether to stick with the direct path, or to draft behind Akamai. We conclude by noting that Akamai is only one of many CDNs; such networks are a great resource that can be use to support an information plane for little to no cost.

References

[1] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante, "Drafting behind Akamai: Travelocity-based detouring," in *ACM SIGCOMM*, Pisa, Italy, September 2006.

[2] Akamai, "Akamai CDN," http://www.akamai.com.

[3] SAVVIS, "Digital island CDN," http://www.savvis.net.

[4] LimeLight Networks, "Limelight networks CDN," http://www.limelightnetworks.com.

[5] Mirror Image, "Mirror image CDN," http://www.mirror-image.net.

[6] A. Shaikh, R. Tewari, and M. Agrawal, "On the effectiveness of DNS-based server selection," in *IEEE INFOCOM*, Anchorage, AK, April 2001.

[7] J. Kangasharju, K. Ross, and J. Roberts, "Performance evaluation of redirection schemes in content distribution networks," *Computer Communications*, vol. 24, no. 2, pp. 207–214, February 2001.

[8] C. Bornstein, T. Canfield, and G. Miller, "Overlay routing networks (Akarouting)," 2002, http://www-math.mit.edu/steng/18.996/lecture9.ps.

[9] P. Gilmore, "OARtech," 2001, http://www.osc.edu/oarnet/oartech/presents/oarnet/11apr2001.ppt.

[10] "Akamai and loral cyberstar alliance," http://www.akamai.com/en/html/about/press/press123.html.
[11] B. Krishnamurthy, C. Wills, and Y. Zhang, "On the use and performance of content distribution networks," in *ACM IMW*, San Francisco, CA, November 2001.
[12] C. Bornstein, T. Canfield, G. Miller, and S. Rao, "Optimal route selection in a content delivery network," US Patent Application 20020163882.
[13] J. Dilley, B. Maggs, J. Parikh, H. Prokop, and R. Sitaraman, "Globally distributed content delivery," *IEEE Internet Computing*, vol. 6, no. 5, pp. 50–58, September 2002.
[14] F. Leighton and D. Lewin, "Global hosting system," US Patent No. 6,108,703.
[15] R. Mahajan, "How Akamai works?" http://www.cs.washington.edu/homes/ratul/akamai.html.
[16] North American Network Operators' Group, "NANOG mailing list," http://www.nanog.org/mailinglist.html, 1999,2000.
[17] "Planetlab," http://www.planet-lab.org/.
[18] K. Johnson, J. Carr, M. Day, and M. Kaashoek, "The measured performance of content distribution networks," in *WCW*, Lisbon, Portugal, May 2000.
[19] A. Medina, M. Allman, and S. Floyd, "Measuring the evolution of transport protocols in the Internet," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 2, pp. 37–52, April 2005.
[20] A. Habib and J. Chuang, "A measurement-based analysis of residential multihoming," in *IEEE INFOCOM, poster session*, Miami, FL, March 2005.
[21] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Reno performance: A simple model and its empirical validation," *IEEE/ACM Transactions on Networking*, vol. 8, no. 2, pp. 133–145, April 2000.
[22] "Kazaa," http://www.kazaa.com/.
[23] K. Gummadi, S. Saroiu, and S. Gribble, "King: Estimating latency between arbitrary Internet end hosts," in *ACM IMW*, Marseille, France, November 2002.
[24] S. Gadde, J. Chase, and M. Rabinovich, "Web caching and content distribution: a view from the interior," in *WCW*, Boston, MA, June 2000.
[25] M. Koletsou and G. Voelker, "The Medusa proxy: A tool for exploring user-perceived web performance," in *WCW*, Boston, MA, June 2001.
[26] B. Krishnamurthy and C. Wills, "Analyzing factors that influence end-to-end web performance," in *WCW*, Amsterdam, Netherlands, April 2000.
[27] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy, "An analysis of Internet content delivery systems," in *USENIX OSDI*, Boston, MA, December 2002.
[28] J. Jung, E. Sit, H. Balakrishnan, and R. Morris, "DNS performance and the effectiveness of caching," *IEEE/ACM Transactions on Networking*, vol. 10, no. 5, pp. 589–603, October 2002.
[29] Z. Mao, C. Cranor, F. Douglis, M. Rabinovich, O. Spatscheck, and J. Wang, "A precise and efficient evaluation of the proximity between web clients and their local DNS servers," in *USENIX Annual Technical Conference*, Monterrey, CA, June 2002.
[30] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of Internet path selection," in *ACM SIGCOMM*, Vancouver, British Columbia, September 1999.
[31] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *ACM SOSP*, Alberta, Canada, October 2001.
[32] C. Tang and P. K. McKinley, "A distributed multipath computation framework for overlay network applications," Michigan State University, Tech. Rep. MSU-CSE-04-18, May 2004.
[33] A. Nakao, L. Peterson, and A. Bavier, "A routing underlay for overlay networks," in *ACM SIGCOMM*, Karlsruhe, Germany, August 2003.
[34] K. Gummadi, H. Madhyastha, S. Gribble, H. Levy, and D. Wetherall, "Improving the reliability of Internet paths with one-hop source routing," in *USENIX OSDI*, San Francisco, CA, December 2004.
[35] Y. Chen, D. Bindel, H. Song, and R. H. Katz, "An algebraic approach to practical and scalable overlay network monitoring," in *SIGCOMM*. ACM, Aug. 2004, pp. 55–66.
[36] T. Fei, S. Tao, L. Gao, and R. Guerin, "How to select a good alternate path in large peer-to-peer systems?" in *IEEE INFOCOM*, Barcelona, Spain, April 2006.
[37] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the Internet," in *ACM SIGCOMM*, August 2003.
[38] M. Wawrzoniak, L. Peterson, and T. Roscoe, "Sophia: An information plane for networked systems," November 2003.
[39] P. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan, "IrisNet: an architecture for a world-wide sensor web," *IEEE Pervasive Computing*, vol. 2, no. 4, 2003.
[40] H. R, J. Hellerstein, N. Boona, T. Loo, S. Shenker, and I. Stoica, "Querying the Internet with PIER," in *Proc. of VLDB*, September 2003.
[41] H. V. Madhyastha, T. Isdal, michael Piatek, C. Dixon, T. Anderson, A. Kirshnamurthy, and A. Venkataramani, "iPlane: an information plane for distributed systems," in *USENIX OSDI*, November 2006.
[42] S. Savage, "Sting: a TCP-based measurement tool," in *USENIX Annual Technical Conference*, Boulder, CO, October 1999.

Ao-Jan Su is a Ph.D. student in the Department of Electrical Engineering and Computer Science at Northwestern University. He received his B.S. degree from National Tsing-Hua University Taiwan in 1998 and M.S. degrees from New York University in 2002. His research interests include network measurement, content distribution networks, routing, and protocols for the Internet.

David Choffnes is a Ph.D. student in the Department of Electrical Engineering and Computer Science at Northwestern University. He received his B.A. degrees from Amherst College in 2002 and M.S. degree from Northwestern in 2006. Choffnes has coauthored textbooks on the topics of operating systems and programming languages, and his current research interests include the design and implementation of large-scale distributed systems in the Internet and in vehicular ad-hoc wireless networks.

Aleksandar Kuzmanovic is an assistant professor in the Department of Electrical Engineering and Computer Science at Northwestern University. He received his B.S. and M.S. degrees from the University of Belgrade, Serbia, in 1996 and 1999 respectively. He received the Ph.D. degree from Rice University in 2004. His research interests are in the area of computer networking with emphasis on design, security, analysis, theory, and prototype implementation of protocols and algorithms for the wired and wireless Internet.

Fabián E. Bustamante did his undergraduate studies in the Universidad Nacional de la Patagonia San Juan Bosco, Argentina. He received his M.S. and Ph.D. in computer science from the Georgia Institute of Technology in 1997 and 2001, respectively. Bustamante is currently an assistant professor of computer science in the Department of Electrical Engineering and Computer Science at Northwestern University. His research interests are in the design and implementation of large-scale distributed systems, in both wired and wireless environments. He is a member of the IEEE Computer Society, the ACM, and USENIX.