Figure 2: A typical Google map snapshot of active DipZoom measurement points.

obtains the content from the origin site, forwards it to the client, and stores it for future use.

In fact, Akamai employs a two-level DNS system (not shown in the figure for simplicity): the centralized high-level servers return NS-type responses to queries, redirecting them to nearby low-level DNS servers, and the latter perform the actual hostname-to-IP resolution. The actual platform architecture involves addressing many other complex issues. For example, Sherman et al. describe Akamai's sophisticated configuration management system [20].

4. METHODOLOGY

This section describes the major experimental approaches we used to conduct our study.

4.1 Edge Server Discovery

Most CDN investigations involve the discovery of the CDN's edge servers [11, 5, 21]. The basic technique for edge server discovery is well established and involves simply identifying and resolving an outsourced hostname. For example, nslookup on "images.amazon.com" or its canonical name "a1794.l.akamai.net" will usually return at least two IP addresses of edge servers.

The challenge, however, arises when one needs to harvest a large collection of edge servers since this requires hostname lookups resolutions from different geographical locations. To avoid the complexities of gaining access to and communicating with multiple hosts, we utilized DipZoom, a peer-to-peer Internet measurement platform, for this purpose [7, 26]. DipZoom has a large number of measurement points around the world, and it allows global experiments to be implemented as local java applications, without the need to explicitly interact with the individual measurement points. While the available DipZoom measurement points vary in time, there are typically more then 400 active MPs available, mostly on PlanetLab nodes but also on some academic and residential hosts. As an indication of geographical coverage, Figure 2 shows a typical Google map snapshot of active DipZoom peers cropped from [7].

In our discovery process, we compiled canonical names outsourced to Akamai by downloading and examining Web page sources of the 95 Akamai customers listed on the Akamai's Web site. We then periodically (twice a week) performed DNS resolution of these names from a large number of network locations over a period of 13 weeks. As the result, we harvested just under 12,000 Akamai edge servers, of which 10,231 servers were pingable at the end of the discovery process. By clustering these edge servers by city[1] and autonomous system, we conservatively estimate we discovered at least 308 locations. The real number is likely higher as the discovered edge servers represented 864 distinct /24 prefixes. Clearly, our discovered portion of Akamai platform represents only a subset of the Akamai's servers and locations; however, as we discuss in Section 6.1, this does not impact our conclusions.

4.2 Overriding CDN Edge Server Selection

In assessing CDN's performance, our performance metric is the effective throughput of the page download as reported by the curl tool [6][2]. To measure download performance from a particular edge server rather than the server of Akamai's choosing, we need connect to the desired edge server directly using its raw IP address rather than the DNS hostname from the URL. We found that to trick an arbitrary Akamai's edge server into processing the request, it is sufficient to simply include the HTTP host header that would have been submitted with a request using the proper DNS hostname.

For example, the following invocation will successfully download the object from a given Akamai edge server (with IP address 206.132.122.75) by supplying the expected host header through the "-H" command argument:

```
curl -H Host:ak.buy.com \
     "http://206.132.122.75/db_assets
     /large_images/093/207502093.jpg"
```

4.3 Controlling Edge Server Caching

Some of our experiments require forcing the HTTP requests to be fulfilled from the origin server and not from the edge server cache. Normally, requesting a cache to obtain an object from the origin server could be done by using HTTP's Cache-Control header. However, as we will show shortly, Akamai's edge servers do not honor the Cache-Control header in client requests [23]. To manipulate the edge server to obtain the content from the origin, we exploit the following observation. On the one hand, modern caches use the entire URL strings, including the search string (the optional portion of a URL after "?") as the cache key. In particular, a request for foo.jpg?randomstring will be forwarded to the origin server because the cache is unlikely to have previously stored an object with this URL. On the other hand, origin servers ignore unexpected search strings in otherwise valid URLs. Thus, the above request will return the valid foo.jpg image from the origin server.

To verify this technique, we performed a series of downloads from "planetlab1.iii.u-tokyo.ac.jp" of an Amazon object "http://g-ec2.images-amazon.com/images/G/01/nav2/gamma/n2CoreLibs/n2CoreLibs-utilities-12475.js" using edge server 60.254.185.89. By selecting the client and edge server that are close to each other but likely distant from the origin server (both the client and the edge server are in Japan while the origin is likely to be in the US), we hope to be

[1]We used the GeoIP database from MaxMind [14] (commercial version) to obtain server's geographical information. GeoIP was able to map 98.13% of our discovered edge servers.

[2]We initially used wget for this purpose but encountered a persistent accuracy problem with wget on some Planet Lab nodes.

	Target & Parameter	Throughput
1	/foo.js?rand1	43 KB/s
2	/foo.js?rand1	7750 KB/s
3	/foo.js?rand1	5300 KB/s
4	/foo.js?rand2	61 KB/s
5	/foo.js?rand2	8000 KB/s
6	/foo.js?rand2	4850 KB/s
7	no-cache, /foo.js?rand2	5070 KB/s
8	no-cache, /foo.js?rand2	6780 KB/s
9	no-cache, /foo.js?rand3	56 KB/s

Table 1: Initial vs. repeat download performance of an object with an appended random search string.

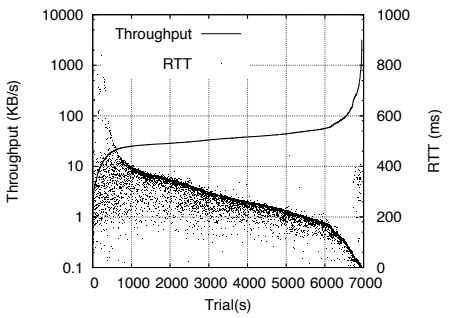


Figure 3: Relation between download throughput and RTT.

able to distinguish download from cache vs. from the origin by the performance difference.

Table 1 lists the throughput of the download series (the long object URL is replaced with "foo.js" for convenience). Download 1 is a non-cache download and shows poor performance. Downloads 2 and 3 are the downloads with the same random string; they exhibit distinctly higher performance, reflecting the fact that these requests are fulfilled from the cache, which stored this URL as the result of the first download. The downloads 4-6 confirm this behavior with a different random string. However, downloads 7 and 8, using the previously seen URLs with a no-cache header, produce similarly high performance, indicating they are processed from the cache despite the no-cache directive. Yet download 9, with the no-cache directive and a new random string, exhibits performance comparable to downloads 1 and 4 and indicative of non-cache downloads. This shows that the No-Cache directive of the HTTP Cache-control header in client requests does not affect caching by Akamai.

In summary, we can precisely control the caching behavior of Akamai's edge servers by appending search strings to image URLs. The first appearance of a string will cause the download to occur from the origin server while the subsequent downloads of the same string from the same edge server will occur from the edge server cache. A series of retrievals of the same content with different random strings will cause the edge server to obtain the original copy every time.

4.4 Assessing Client-Side Caching Bias

Most of our experiments below involve comparing download performance from different Akamai locations. A crucial question is a possible bias introduced by transparent caching potentially used by the ISPs through which our measuring points connect to the Internet. Indeed, a cached download would then be performed from the cache regardless to which Akamai location we ostensibly direct the HTTP request.

To estimate the extent of transparent cache use by the ISPs utilized by DipZoom MPs, we consider how HTTP download performance from a given Akamai edge server depends on its ping round-trip latency. If transparent caching is widely used, we would see no dependency.

To study these correlations, we perform a set of four curl downloads, five pings, and a traceroute from each measuring point to various edge servers. For HTTP download performance, we average the download throughputs reported by the last three curls (the first curl ensures that the object is in the edge server cache); for ping RTT, we average the RTTs from the five pings. We use an outsourced cacheable 437,688B Amazon object for curl downloads.

We used 385 measuring points and 20 widely distributed edge servers in this experiment. To ensure that target edge servers are widely distributed across the world, we clustered all the discovered edge servers into twenty clusters based on the estimated distance between the servers (see Section 6) and selected centers of the clusters as our targets.

Figure 3 plots the values of the ping RTT and curl download throughput in each trial, sorted in the order of the increasing throughput. (The total number of trials is less than 385×20 because not all measurements were successful.) The figure shows clear dependency of the download performance on ping distance. Quantitatively, the two metrics show Spearman's rank correlation coefficient of -0.8074 indicating a strong dependency and thus no significant use of client-side ISP caching.

In fact, we later discovered a direct method to detect a transparent cache. We deploy our own Web server with a cache-able object and request it twice from every measuring point. We then examine the server' access log. If the downloads from a given MP are filtered through an ISP cache, the log would contain only one request from this MP (the other would be terminated by the ISP cache); otherwise the log would contain both requests. We tested 527 measurement points (as many as we could assemble) and found only three[3] with a single request in our log. Thus, the overwhelming majority (99.43%) of MPs had no transparent caches between them and the Web servers. While this test was conducted a few months later than the rest of our study, the two experiments described here give us a high level of confidence that client-side caching does not bias our findings.

4.5 Measuring Edge Server Performance

In Sections 5 and 6.1, we use download performance as the "goodness metric" of a given edge server, and we express it as effective throughput, which is the object size divided by the download time. Given that the performance of downloads of different-sized objects is predominantly affected by different network characteristics (bandwidth for large objects and RTT for small objects), we verify our main findings on objects of a wide size range (from 10K to 400K). In addition, Section 6.3 uses round-trip time from the client as the goodness metric of a given server.

[3]These MPs were 137.132.250.12 (nusnet-250-12.dynip.nus.edu.sg, Singapore), 132.252.152.193 (planetlab1.exp-math.uni-essen.de, Germany, Essen), and 147.126.95.167 (unknown hostname, Chicago).

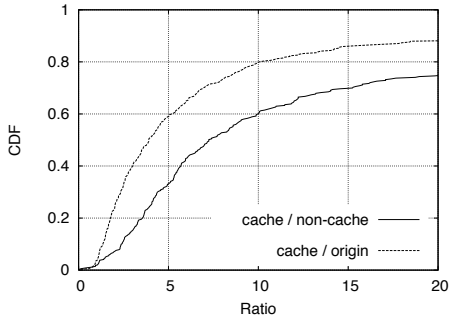


Figure 4: The performance benefits of Akamai delivery.

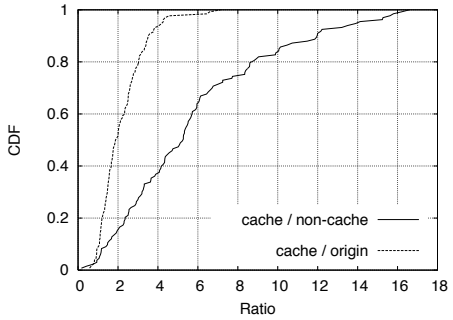


Figure 5: The residential client performance benefits of Akamai delivery.

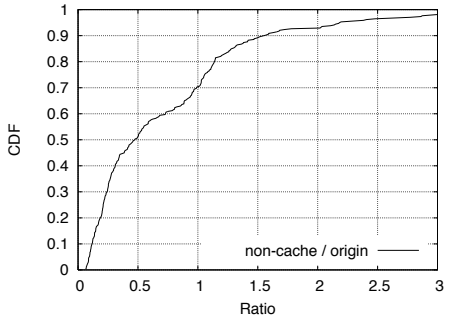


Figure 6: The comparison of no-cache download through Akamai and download from origin server.

5. PERFORMANCE OF AKAMAI CDN

This section considers how well the existing Akamai platform fulfills its mission of accelerating content delivery. First, we study how it affects the end-user download performance, then consider the quality of its server selection.

5.1 Does a CDN Enhance Performance?

In general, a content delivery network can improve Web performance in two ways: by providing overload protection to Web sites and by delivering content to users from nearby locations. In this study, we are focusing on the latter aspect.

To answer this question precisely, one would need to compare download performance from a CDN with the direct download of the same object from the origin server. Unfortunately, we found that the origin servers hosting Akamai-delivered content are usually hidden from the Internet users: our attempts to request this content from the origin sites that host the container pages were not successful.

Thus, we resort to two estimates of the download performance from the origin servers. First, we estimate it by the download time of the Akamai-delivered content when we prevent caching at the edge servers. This actually measures the Akamai cache miss performance and hence adds the miss penalty to the true value. But it can indicate the performance of the direct download, assuming Akamai infrastructure is highly optimized. Second, we estimate the origin server performance by the download performance of a different object of similar size, which is not outsourced to CDN delivery, from the same Web site. Even if the non-outsourced object is served from a different server than the one used by Akamai on a miss, the performance of this download indicates the performance that the Web site can achieve without resorting to an external CDN.

We used 377 measuring points for these experiments. For the CDN-delivered content, we use an outsourced Amazon object with size 50K, and we use the methodology in Section 4.3 to control its caching. For the download from the origin, we found a non-outsourced static page[4] of 55K bytes on the Amazon website. (We assume the page is static by the fact that multiple downloads from different vantage points resulted in the same content.) Obviously, the results in this subsection are strictly applicable only to Amazon, as another customer may have a differently provisioned origin site with different performance. However, they offer a specific data point for a large, and presumably well-provisioned, content provider.

[4]http://72.21.206.5/gp/help/customer/display.html

Figure 4 plots the cumulative distribution functions of the cache-to-no-cache and cache-to-origin throughput ratios in all the collected measurements. In both cases, values over 1 mean the CDN improves performance over direct delivery (under the corresponding estimate of the direct delivery performance) and values below 1 indicate the opposite. The figure indicates that a CDN promises a significant performance improvement under both estimates of the direct delivery performance. CDN delivery outperforms both no-cache and origin delivery in 98% and 96% of cases respectively. Furthermore, in 67% and 41% of the cases, the CDN delivery is at least five times faster than no-cache and origin delivery respectively. A possible reason for such dramatic improvement could be because most of our measuring points are well connected to the Internet and do not experience the last mile bottleneck.

Therefore, to see the benefits for residential clients, we reproduce, in Figure 5, the same results but only for measuring points with maximum download bandwidth less than 1.5Mbps – typical residential connectivity. For these measuring points, the performance benefits drop significantly. CDN delivery is now at least five times faster that origin delivery only in 2.3% of the cases. However, CDN delivery still improves the download performance of residential clients: the CDN delivery outperforms no-cache in 95.5% of cases and origin delivery in 91% of cases.

Finally, the gap between the two curves in both figures indicates the Akamai global cache miss penalty, i.e., the overhead the edge servers add for a request when the requested object has to be fetched from the origin server. We quantify this penalty in Figure 6, which plots the CDF of the no-cache-to-origin throughput ratio. It shows that in 70% of the cases, origin downloads are faster than no-cache down-

loads, and in 50% of the cases they are at least twice as fast. This indicates that Akamai miss penalty can be significant.

A potential limitation of the experiment in this subsection is that it considers performance of a single Web object and not entire page downloads. Pages typically include multiple embedded objects, which may affect direct and CDN-accelerated downloads differently. Indeed, depending on the setup of the CDN service, the browser can download the initial container HTML object directly from the origin site and the embedded objects from the CDN's edge server. Thus, the CDN delivery could entail an extra DNS resolution (to resolve host names of embedded URLs) and an extra TCP connection establishment (to establish connections to both the origin and edge server). However, this should not materially affect our conclusions because these effects are amortized over the embedded objects on a page and often over several pages in a session (if the user accesses these pages in short succession so that cached DNS responses and persistent connections remain valid between page accesses). Akamai allows clients to cache DNS responses for 20 seconds (although many use them far longer [15]). We also probed around 1000 Akamai edge servers for persistent connection support and found they maintain them for unusually long time - 500 seconds after a request (compared to Apache's default of 15s). This provides ample opportunity for connection reuse.

5.2 How Good Is Akamai Server Selection?

When a client initiates the download to a content cached by Akamai, the best edge server for this particular client and content is selected dynamically. This section investigates how good is the Akamai selection algorithm from the client performance perspective.

To answer this question, we selected one customer, Amazon, and measured the download throughput of an Akamai-delivered object of 50K bytes from 391 measuring points. Out of 10,231 edge servers we discovered, 430 servers were discovered for Amazon. At each of the 391 measuring points, 80 edge servers are randomly selected from these 430 servers, and the object is downloaded from the Akamai-selected edge server and from each of the 80 alternative edge servers, using the methodology of Section 4.2 to download from a particular edge. Because the entire experiment from a given measuring point takes considerable time, we always perform a pair of measurements one right after the other, one for the Akamai-selected server, and one for a given alternative edge server. Note that the Akamai-selected server may change from one pair to the next. Finally, we use the techniques of Section 4.3 to ensure that the edge server delivers the object from its cache in each case.

Figure 7 shows, for each measuring point, the percentage of time that the Akamai-selected server performed better than the alternative server, in the increasing order of this metric. For example, at measuring point number 200, it shows that roughly 80% of the random 80 edge servers were outperformed by the Akamai-selected server. The server selection was less successful for the 199 measurement points numbered lower and more successful for the the 191 measuring points numbered higher. Overall, this experiment confirms an early limited study [9] that CDNs rarely select the best edge server but successfully avoid the worst ones. Indeed, in roughly 75% of the MPs, the Akamai-selected server outperformed half of the alternatives.

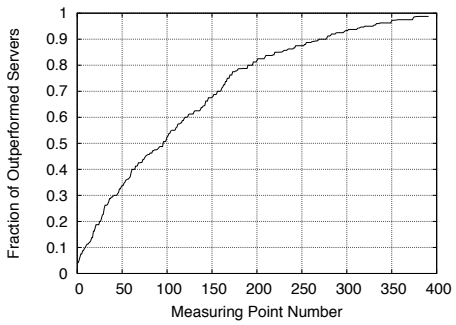


Figure 7: The fraction of servers outperformed by the Akamai-selected server.

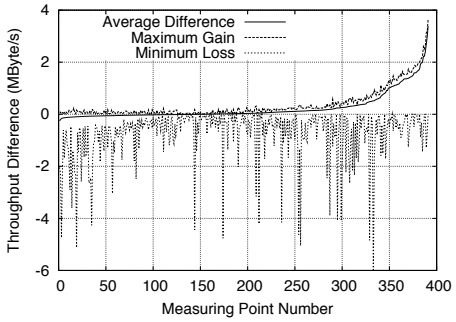


Figure 8: Download throughput difference between Akamai-selected and an alternative edge server

Figure 7 shows how many times Akamai-selected servers perform better or worse than the alternatives but does not say by how much. To provide this information we plot on Figure 8 the average difference, the maximum gain, and the maximum loss of throughput of the Akamai-selected server vs. the alternatives. The average difference is taken over all 80 alternative servers; negative values indicate the alternative servers performed better on average and positive values correspond to the advantage of the Akamai-selected server. Maximum gain and maximum loss are the throughput differences of the Akamai-selected server over, resp., the slowest and fastest alternative server. If positive, the maximum gain shows the advantage of the Akamai-selected server over the worst alternative. Similarly, if negative, the maximum loss shows the maximum penalty vs. the best alternative.

The "average difference" curve shows that there are 147 measuring points where Akamai-selected server delivers on average inferior performance, by at most 255 KB/s for measuring point 1. However, Akamai-selected servers have superior performance in average on 244 measuring points with up to 3.38 MB/s average advantage. Interestingly, although Akamai-selected servers have superior performance on average, the "maximum loss" curve shows that, in a number of cases, the best alternative servers outperform Akamai-selected servers significantly, by up to 6.35 MB/s.

In summary, Akamai usually makes good server selection decisions, but there is a substantial room for further improvements.

6. PERFORMANCE OF CONSOLIDATED AKAMAI CDN

Akamai attempts to deliver content to users from nearby servers by placing edge server in a large number of network locations. The question we pose is: how much would it cost in performance if the number of data centers were reduced significantly?

We first describe our technique for data center consolidation, then present a study using measurements from DipZoom measurement points, and conclude with a live study involving real Internet users. Both studies show that a considerable consolidation is possible without a noticeable effect on performance.

6.1 Data Center Consolidation

Our methodology for studying a hypothetical consolidated CDN is as follows. We first group edge servers that we believe are close to each other into hypothetical consolidated data centers, which we refer to as *big clusters*. We then "place" each big cluster into the location of a central server in the cluster called the representative of the cluster. To this end, for a given client, we replace the server selected by Akamai, S_{akam} with the representative of the cluster to which S_{akam} belongs. In other words, we assume that all clients that would have been sent to any server in a given big cluster in the existing platform, will be served from the cluster center in the consolidated case. We then consider performance implications of this replacement by comparing the performance of the downloads from both servers.

We would like to stress again that our study only considers the implication of CDN consolidation on the proximity of clients to data centers: the aggregate CDN capacity, both in terms of network bandwidth and server capacity, is orthogonal to the number of data centers as fewer locations can be compensated by higher processing and connectivity capacity at each data center. Because our probes impose trivial load on Akamai servers, the performance of our downloads reflect the proximity between servers and clients under normal load. In fact, any server load differences in individual server-pair comparisons should work *against* platform consolidation because Akamai avoids overloaded servers in its server selection while we use cluster centers regardless of their load.

To cluster edge servers, we start by estimating the pairwise network distances between all the servers using a recently proposed dynamic triangles method [25], and then group nearby servers into a predefined number of big clusters by applying a hierarchical clustering algorithm to the resulting distance matrix. (We use the so-called *hierarchical clustering with complete linkage* method, following by the *cut-the-tree* procedure, both provided by the R software [18] for this computation.) We could equally use other techniques, such as network-aware clustering [10]. However, our goal is to show that we can consolidate large numbers of servers into fewer locations, and how we select servers for consolidation is immaterial as long as we find the performance of the consolidated platform comparable. In other words, imperfect clustering makes our finding conservative: Better clustering could lead to a better-constructed consolidated platform with even fewer data centers.

Similarly, a consequence of incomplete discovery of Akamai's platform is that we have fewer potential locations for consolidated data centers (indeed, our methodology only chooses between discovered edge servers as potential consolidated locations). This makes our finding that significant consolidation is possible, again, more conservative. In other words, the incomplete discovery does not undermine our findings regarding the consolidation of the locations that we did discover.

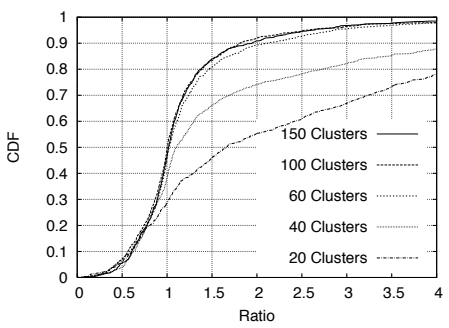


Figure 9: The performance of a consolidated Akamai platform with different number of data centers.

6.2 DipZoom Experiment

To judge the performance of the hypothetical consolidated Akamai platform, we compare the performance of downloads from the current and consolidated platforms using DipZoom measurement points. We had 412 measurement points in this experiment. From each measurement point, we downloaded the object of a given size from the Akamai-selected server and from the center of its cluster in the consolidated platform. The center is represented by a randomly selected available server from the five closest servers to the center of the cluster[5].

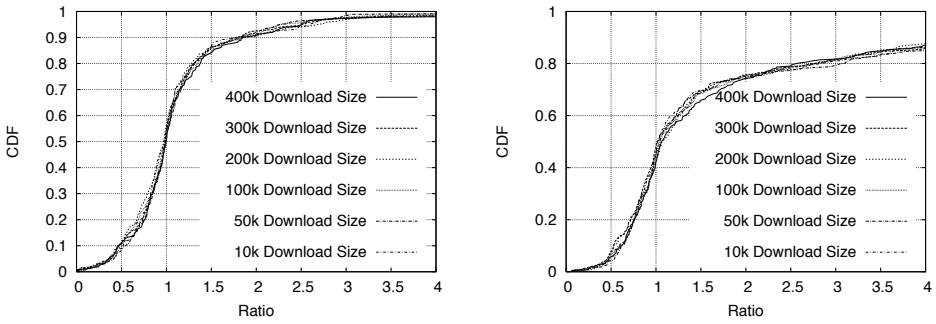
To avoid the bias from changing network conditions, we perform each pair of downloads in immediate succession. Further, we pre-request each object from both servers to ensure the server is delivering the object from its cache, thus excluding the possibility of skewing results by the cache-miss penalty. Having placed the object into the servers' cache, we perform three downloads from either server and take the average as its download performance. We compare the performance of the existing and consolidated Akamai platforms by the ratio between the download throughput in both platforms. The performance ratio more than 1 means Akamai's existing configuration yields better download performance than the consolidated configuration and vice versa.

We grouped the $10,231$ edge servers into 150, 100, 60, 40, and 20 data centers, thus considering the varying degree of consolidation. To check if our conclusions might depend on the size of downloaded objects, we controlled the size of the downloads precisely by finding a large (over 400K) object[6] and specifying an appropriate "Range" header in our HTTP request. We verified that Akamai servers honor range requests.

Figure 9 presents the CDF of the download throughput ratios of the existing-to-consolidated configurations. The figure reflects data points obtained by downloading an out-

[5]We choose from five closest servers instead of always using the centroid node to reduce an undue effect of a single server on the entire experiment.

[6]http://<...>/images/G/01/digital/video/preview/Player._V16565163_.swf



(a) 60 Data Centers (b) 40 Data Centers

Figure 10: The performance of a consolidated Akamai platform with different target object size.

sourced Amazon object of size 150K, 100K, 50K, and 10K from 412 measurement points world-wide. The curves reflect 5522 ratios (some downloads were unsuccessful).

As seen in Figure 9, consolidating edge servers into 150, 100, and 60 data centers does not cause noticeable performance degradation: the consolidated and current platforms show performance advantage over each other with equal probability. Only when we get down to 40 and 20 data centers, does the original platform start outperforming the consolidated configuration – 60% and 70% of the time for 40 and 20 data centers, respectively. Furthermore, as Figure 10 shows, these results are largely independent of the target object size.

The above experiment reflects our mix of measurement points, which are skewed towards well-connected hosts. Because CDNs are often used by high-volume consumer-oriented sites, where users may have lower capacity connections, we consider the performance of consolidated configurations separately for measurement points with different download bandwidth. Specifically, we group the measurement points by the maximum download throughput observed in the course of the experiment of Figure 9.

The results, shown in Figure 11, indicate that the existing Akamai configuration with large number of data centers favors well-connected users. For measurement points with over 6Mbps bandwidth, the existing configuration outperformed consolidated configurations once they get down to 60 data centers. However, the less the bandwidth of the measurement points the less the advantage of the existing configuration. Given this trend, and because CDNs are often used by high-volume consumer-oriented sites, a pertinent question is how the consolidation may affect typical residential users.

Thus, we compare, in Figure 12, the performance of existing vs. consolidated configurations for all measurement points with bandwidth below 1.5Mbps, which is a typical download bandwidth for DSL users. Figure 12 shows that these clients would not see noticeable performance difference if the servers were further consolidated into 40 data centers.

Overall, we conclude that one could consolidate our discovered portion of Akamai platform to 60 data centers, and for typical residential users, even to 40 data centers without noticeable performance penalty. Even compared with our conservative approximation of 308 original data centers, this represents significant consolidation. However, as we mentioned, the real number of Akamai locations represented in our study is likely higher, which makes our findings even more significant.

6.3 A Live Study

The study of the previous subsection measured real downloads but performed them from DipZoom measurement points. In this section, we perform live measurement from a larger number of vantage points and from real Internet users but measure the latency between the users and the edge servers. We built a Web page[7] with an AJAX application that, when loaded, measures latency to the server that Akamai selects for this browser as well as to a list of other Akamai edge servers, and reports the results to us. We requested a mid-size commercial company to embed our special page into a zero-sized frame, and we also embedded it into our own Web pages. As clients access these Web sites, we collect our measurement results.

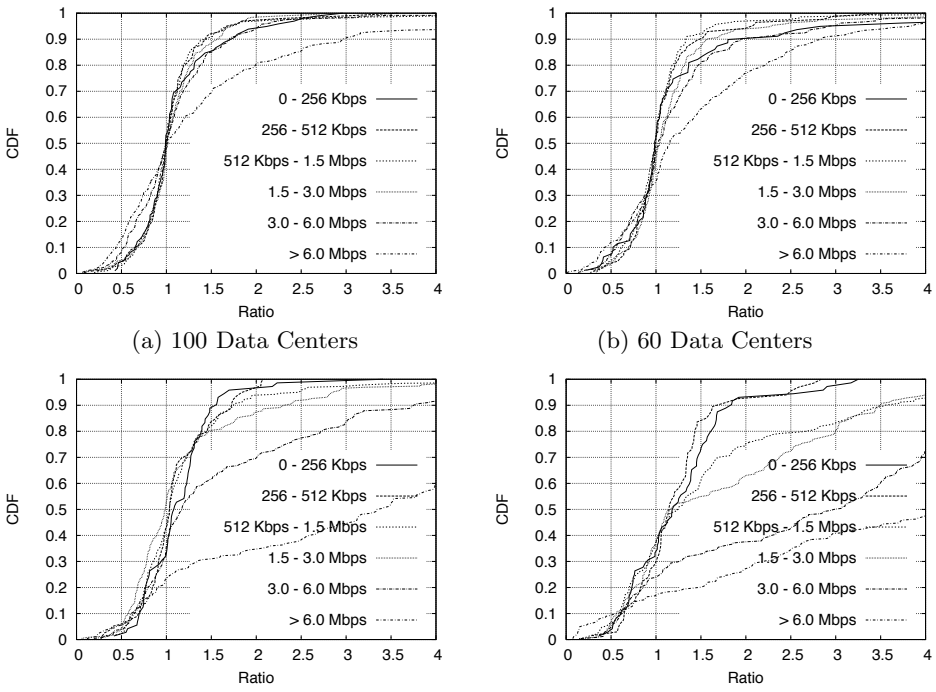
We picked a CNAME (a1694.g.akamai.net, utilized by pcworld.com) which we found is mapped by Akamai to a large number – 979 – of edge servers. These servers represent 343 different /24 networks; using the city+AS heuristic, we estimate them to represent 168 locations.

To compare the current configuration with consolidated configurations, we partitioned the 979 servers into 10, 50 and 100 clusters using our clustering approach. We selected 100 servers from the centers of the 100 clusters and measured the latency from clients to these servers from the AJAX application. To cut down on the number of measurements, we selected representatives of larger clusters among these 100 servers as well (in our clustering, a larger cluster is built as a union of smaller clusters, so we could always find an appropriate server this way). This does not undermine our results because we have a discretion where to place our consolidated data centers.

We could not find a way to pass a custom Host header to requests sent from Javascript, which is necessary to obtain a real object from a non-Akamai-selected edge server (Section 4.2. Thus, we submit a bogus URL that returns "Bad Request" in all cases. We verified with tcpdump that obtaining this response by the client involves two RTTs, allowing us to measure the latency between the client and the edge server involved.

We have collected 24,079 measurements from 2,926 client IP addresses by the time of this writing. According to the GeoIP database, these vantage points cover a wide area, representing 47 US states and 43 foreign countries, and as

[7]Available at http://haddock.case.edu:8000/realdistquery.



(a) 100 Data Centers (b) 60 Data Centers

(c) 40 Data Centers (d) 20 Data Centers

Figure 11: The performance of a consolidated Akamai platform with different download speed.

our statistical analysis will show, are sufficient to derive a meaningful result.

We consider the differences between the measured latencies from each client to its closest consolidated data center and the Akamai-selected server. A negative value indicates that the consolidated platform performed better than the existing Akamai platform in the corresponding observation, while a positive value reflects an advantage of the existing Akamai platform. This assumes that the consolidated configuration performs perfect data center selection and thus should be viewed as an upper bound of the consolidation benefits. However, we note that fewer farther-apart locations make server selection easier.

To avoid using correlated measurements in the analysis, we average all the measurements from the same client, so that each client will contribute only one data point to our analysis. Further, to remove possibly correlated measurements from the same network in the same locale, we use only one randomly selected client from all the clients with the same city and autonomous system according to GeoIP[8]. This reduced the number of data points for our analysis from 2,926 to 2,029.

To assess the significance of the results, we build confidence intervals for the reported means. Because the distribution of the observations is unknown, we use the nonparametric bootstrap method to estimate the population mean and median and to build the confidence interval for the mean. In particular, we used the bootstrap bias-corrected accelerated (BCa) interval method [16] with 10,000 resampling sets and relied on Matlab-provided functions that implement the core of the method.

[8]Note that the observations should not be averaged in this case because we do not know if in fact these measurements are all correlated.

Table 2 summarizes the results of these measurements, with confidence intervals for the means built for confidence probability 95%. The results show that with both 100 and 50 consolidated data centers, we could still find a closer data center than the Akamai-selected server for a majority of clients, and the average distance to the nearest consolidated data center across all clients is also lower. In fact, the entire confidence interval for the mean distance difference is negative, indicating that the above conclusion is statistically significant, and even the upper limit of the confidence interval indicates the RTT difference of over 40ms for 50 data centers and 70ms for 100 data centers. Only with 10 data centers do we see a mixed result: while the entire confidence interval for the average distance difference is still negative (indicating the distance from clients to the nearest consolidated data center still smaller on average than to the Akamai-selected server in the current platform), the median difference indicates an advantage of the current platform, indicating that current Akamai platform would perform better for a majority of clients. We also note that the median difference between Akamai and consolidated platforms is much smaller than average in all configurations. This says that the average difference is skewed against Akamai by occasional poor server selections, confirming a finding from Section 5.2.

Overall, our results show that even with significant consolidation, better server selection can more than make up for any performance impact from consolidation.

7. CONCLUSION

This paper presents a large-scale performance study of the Akamai CDN. Using DipZoom measurement platform, we were able to discover a large number of Akamai edge servers for our study, and to utilize hundreds of vantage points for our performance measurements. To our knowl-

Configuration	Best in 10	Best in 50	Best in 100
Sample Median	16.00	−5.00	−18.40
Sample Mean	−15.66	−54.43	−81.53
Bootstrap Median	15.954417	−5.013368	−18.260679
Bootstrap Mean	−15.695077	−54.313225	−81.697586
95% conf. interval of the mean	[−33.816798, −5.129345]	[−72.209521, −43.798741]	[−99.081914, −70.692385]

Table 2: The difference of RTT distance (in milliseconds) from clients to the nearest data center in a given consolidated platform and to the Akamai-selected server in the current platform (live clients).

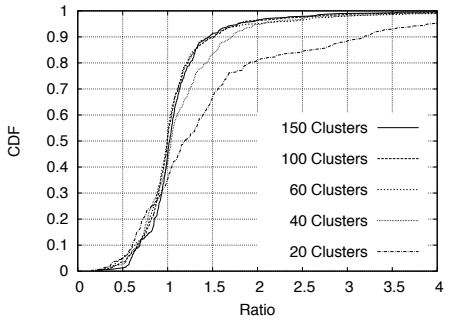


Figure 12: The performance of a consolidated Akamai platform for residential speed links.

edge, ours is the first study to provide an independent direct estimate of the performance improvement of Akamai-accelerated downloads. We further studied the quality of Akamai server selection at a large scale. A central to this paper is an evaluation, from the performance perspective, of the possibility of consolidating Akamai's platform into fewer large data centers. We found that quite significant consolidation is possible without appreciably degrading the platform performance.

8. REFERENCES

[1] Akamai Technologies. http://www.akamai.com/html/technology/index.html.

[2] Akamai Technologies. Facts and Figures. http://www.akamai.com/html/about/facts_figures.html.

[3] Akamai Technologies. Perspectives. http://www.akamai.com/html/perspectives/index.html.

[4] A. Biliris, C. Cranor, F. Douglis, M. Rabinovich, S. Sibal, O. Spatscheck, and W. Sturm. CDN brokering. In *6th Int. Web Caching Workshop*, 2001.

[5] C. Canali, V. Cardellini, M. Colajanni, and R. Lancellotti. Evaluating user-perceived benefits of content distribution networks. In *Int'l Symp. on Perf. Eval. of Comp. and Telecomm. Sys. (SPECTS)*, 2004.

[6] cURL - tool for transferring files with url syntax. http://curl.haxx.se/.

[7] Dipzoom - deep internet performance zoom. http://dipzoom.case.edu.

[8] Intelligent content distribution service. AT&T Inc. http://www.business.att.com/service_fam_overview.jsp?repoid=ProductSub-Category&repoitem=eb_intelligent_content_distribution&serv_port=eb_hosting_storage_and_it&serv_fam=eb_intelligent_content_distribution&segment=ent_biz.

[9] K. Johnson, J. Carr, M. Day, and M. Kaashoek. The measured performance of content distribution networks. In *5th Web Caching Workshop*, 2000.

[10] B. Krishnamurthy and J. Wang. On Network-Aware Clustering of Web Clients. In *The ACM SIGCOMM*, Aug. 2000.

[11] B. Krishnamurthy, C. Wills, and Y. Zhang. On the use and performance of content distribution networks. In *The First ACM SIGCOMM Internet Measurement Workshop*, pages 169–182, 2001.

[12] Limelight Networks brings dynamic streaming to the web. http://blog.llnw.com/2009/08/limelight-networks-brings-dynamic-streaming-to-the-web/.

[13] Limelight Neworks. http://www.limelightnetworks.com/platform/cdn/.

[14] MaxMind. http://www.maxmind.com.

[15] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, and S. Seshan. On the responsiveness of dns-based network control. In *ACM IMC*, pages 21–26, 2004.

[16] I. Poese, B.Frank, B. Ager, G. Smaragdakis, and A. Feldmann. Improving content delivery using provider-aided distance information. In *ACM IMC*, 2010.

[17] Rapid Edge Content Delivery Network. http://www.rapidedgecdn.com/.

[18] The R project for statistical computing. http://www.r-project.org.

[19] Savvis Inc. http://www.savvis.net/corp/Products+Services/Digital+Content+Services/Content+Delivery+Services/.

[20] A. Sherman, P. A. Lisiecki, A. Berkheimer, and J. Wein. Acms: the akamai configuration management system. In *NSDI*, pages 245–258, 2005.

[21] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante. Drafting behind akamai (travelocity-based detouring). In *SIGCOMM*, pages 435–446, 2006.

[22] A.-J. Su and A. Kuzmanovic. Thinning akamai. In *The 8th ACM IMC*, pages 29–42, Oct. 2008.

[23] S. Triukose, Z. Al-Qudah, and M. Rabinovich. Content delivery networks: Protection or threat? In *ESORICS*, pages 371–389, 2009.

[24] S. Triukose, Z. Wen, and M. Rabinovich. Content delivery networks: How big is big enough? (Poster paper). In *ACM SIGMETRICS*, 2009.

[25] Z. Wen and M. Rabinovich. Network distance estimation with dynamic landmark triangles. In *ACM SIGMETRICS*, pages 433–434, 2008.

[26] Z. Wen, S. Triukose, and M. Rabinovich. Facilitating focused internet measurements. In *ACM SIGMETRICS*, pages 49–60, 2007.