A Methodology for Estimating Interdomain Web Traffic Demand

Anja Feldmann★, Nils Kammenhuber★,≍,
Olaf Maennel★, Bruce Maggs◇,•,⊙,⋈,≻, Roberto De Prisco△,•, Ravi Sundaramþ,•

★ Technische Universität München {feldmann,hirvi,olafm}@net.in.tum.de • Akamai Technologies {bmm,roberto,koods}@akamai.com
◇ Carnegie Mellon University △ Università di Salerno þ Northeastern University
Works partly funded by grants: ≍ DFG-Schwerpunkt 1126 ⊙ NSF ITR Award ANI–0331653 ⋈ NSF ITR Award CCR–0205523 ≻ US ARO Award DAAD19-02-1-0389

ABSTRACT

This paper introduces a methodology for estimating interdomain Web traffic flows between all clients worldwide and the servers belonging to over one thousand content providers. The idea is to use the server logs from a large Content Delivery Network (CDN) to identify client downloads of content provider (i.e., publisher) Web pages. For each of these Web pages, a client typically downloads some objects from the content provider, some from the CDN, and perhaps some from third parties such as banner advertisement agencies. The sizes and sources of the non-CDN downloads associated with each CDN download are estimated separately by examining Web accesses in packet traces collected at several universities.

The methodology produces a (time-varying) interdomain HTTP traffic demand matrix pairing several hundred thousand blocks of client IP addresses with over ten thousand individual Web servers. When combined with geographical databases and routing tables, the matrix can be used to provide (partial) answers to questions such as "How do Web access patterns vary by country?", "Which autonomous systems host the most Web content?", and "How stable are Web traffic flows over time?".

Categories and Subject Descriptors C.2.3 [**Computer Communication Networks**]: Network monitoring

General Terms: Measurement, Management, Analysis, Algorithm

Keywords: Traffic matrix, Web, Traffic demand, Interdomain, Estimation

1. INTRODUCTION

The reliable estimation and prediction of network traffic demands has tremendous utility. Internet Service Providers (ISPs) routinely employ traffic demand matrices for network capacity planning and traffic engineering [1]; demand matrices enable the identification of bottleneck links and the evaluation of failure scenarios. Network security analysts rely on models of normative traffic demands to detect new threats; worms and distributed denial of service (DDoS) attacks often generate pathological traffic patterns. Traffic demand matrices are also a critical input into simulators for the design and development of new protocols and services.

IMC'04, October 25–27, 2004, Taormina, Sicily, Italy.

1.1 Our Contributions

Interdomain traffic estimation is the focus of this paper. An understanding of the complexity of this problem is best gained by contrasting with the case of **intradomain** traffic estimation which can be done in several ways. In theory an ISP can read these demands directly from its routers using tools such as Netflow (in practice the volume of data introduces complications). An alternate approach is to use SNMP to collect link-level load measurements, and then to generate a "reasonable" traffic demand matrix compatible with these measurements [2]. These approaches have proven effective and are used in practice today. Modeling **interdomain traffic demands**, on the other hand, is problematic because no single organization has the authority or the ability to measure all network traffic. An ISP can measure the demands of its clients, and the "transit" traffic that it carries on behalf of other ISPs, but even the largest Tier-1 ISP has been estimated (folklore) to carry less than 7 % of the Internet's traffic.

This paper presents a methodology for estimating a significant part of the interdomain traffic demand: HTTP traffic between over one thousand (mostly) United-States-based content providers and all clients worldwide. Our approach is based on four observations.

1. Content delivery networks (CDNs) deliver a significant fraction of the bytes downloaded by university Web users. In particular, Saroiu et al. [3] observed that about 4.3% of the Web traffic received by clients at the University of Washington between May 28th and June 6th, 2002, was delivered by Akamai.
2. For each HTTP request recorded in a CDN's Web server logs, the same client typically makes several additional requests directly to the content provider's Web servers.
3. For each object served by a CDN, the objects typically served directly by the content provider can be identified by examining traces of Web usage from large groups of users, or by examination of the content provider's Web site.
4. The locations of the content provider's Web servers can be determined with the help of the DNS system and information available from the interdomain routing system.

In this paper, we combine server logs from Akamai's CDN network with HTTP traces extracted from packet traces gathered at several universities to build detailed traffic demand matrices. We provide two types of matrices. *Publisher demand* matrices pair hundreds of thousands of client IP blocks with over one thousand publishers. *Web traffic demand* matrices pair these client blocks

with tens of thousands of IP addresses belonging to publisher and CDN Web servers. For each pair, in either type of matrix, we estimate the rate at which data is transferred to the clients in the block.

1.2 Related Work

The book by Balachander and Rexford [4] contains an excellent survey of Web usage studies. Some studies have focused on understanding user behavior [5, 6, 7, 8], while others have looked at various aspects of changes in content [9] including the effects of these changes on the traffic demands [10]. The effects of such changes especially as imposed on a Tier-1 ISP have been studied by [11, 12, 13, 14]. The impact on end-to-end performance experienced by the users has been explored using both passive [15] and active measurements [16].

As mentioned earlier, there are a variety of approaches for estimating intradomain traffic matrices. This topic has been the topic of intense research over the past three years [17, 2, 18, 13, 12, 19, 20, 11].

A literature survey reveals that intradomain traffic engineering algorithms [21, 1] have been a principal research focus. This is not mere coincidence because a primary input to most *traffic engineering* algorithms is a *traffic demand* matrix.

Very recently a number of schemes for Interdomain traffic engineering have been proposed [22, 23, 24, 25, 26, 27, 28]. However, to the best of our knowledge, there is no good methodology for estimating interdomain traffic demands. It is our understanding that even the question of whether interdomain traffic matrices and intradomain traffic matrices have similar dynamics remains unanswered.

1.3 Outline

The remainder of this paper is organized as follows: in Section 2 we provide background information concerning content delivery networks, and establish terminology for the paper. Section 3 introduces the notions of publisher demand and Web traffic demand. Section 4 discusses how to estimate publisher demands using a CDN. Section 5 explains how we combine logs from a CDN with packet traces to estimate publisher demands, and how we turn publisher demands into Web traffic demands (the details of our implementation are provided in an appendix). A description of the individual data sets we use is given in Section 6 while Section 7 presents initial results obtained by analyzing the spatial and temporal properties of the traffic demands. Finally, in Section 8 we summarize our experience and suggest future research directions.

2. BACKGROUND: CDN S AND TERMINOLOGY

This section presents a brief overview of the process of content delivery with and without content delivery networks (CDNs). We also present a brief dictionary of the terms and abbreviations used in the remainder of the paper.

2.1 Terminology

The following definitions, taken in part taken from the Web Characterization Terminology & Definitions Sheet [29], will serve to clarify the subsequent discussions.

Web site: A collection of interlinked Web objects hosted at the same network location by a set of **origin Web servers**.

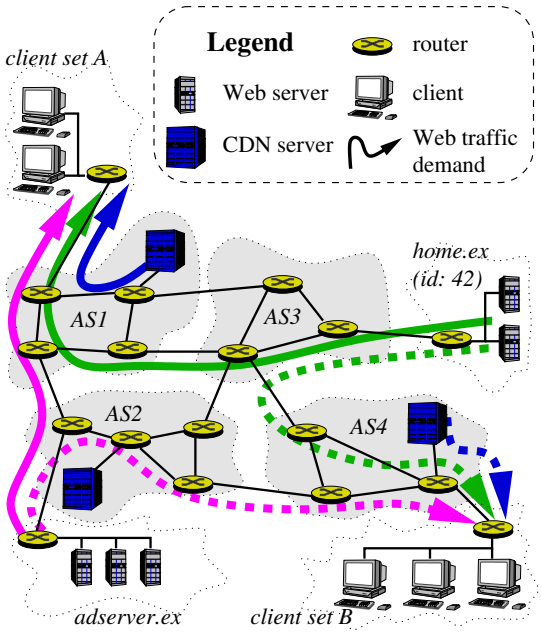


Figure 1: Example of CDN deployment and traffic flows (Web traffic demands).

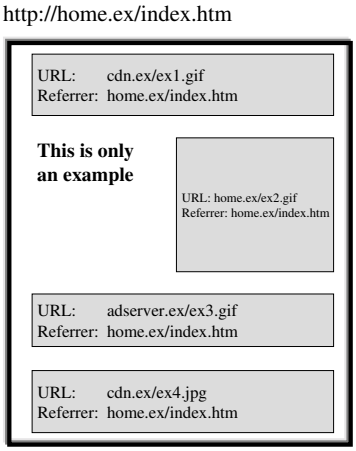


Figure 2: Example Web page with some CDN content.

Supersite: A single, logical Web site that extends over multiple network locations, but is intended to be viewed as a single Web site.

Web site publisher, or just **publisher:** A person or corporate body that is the primary claimant to the rewards or benefits resulting from usage of the content of a Web site. A publisher may distribute his content across multiple Web sites. Publishers are also referred to as content providers.

Content delivery network: An alternative infrastructure operated by an independent service provider on which some parts of a Web site can be hosted.

2.2 Content delivery

The Internet is most commonly used to exchange or access information. This information is typically hosted on origin Web servers. Content Delivery Networks (CDNs) (see, e.g., [30, 31, 32, 33, 34,

35, 3]) are designed to reduce the load on origin servers and at the same time improve performance for the user. Most CDNs have a large set of servers deployed throughout the Internet and cache the content of the original publisher at these servers. Therefore another view of CDNs is that they provide reverse proxy services for content providers, the publishers. In order to take advantage of their distributed infrastructure, requests for data are redirected to the "closest" cache server. Intelligent redirection can reduce network latency and load (and therefore network congestion) improving response time. CDNs differ in their approach to redirecting traffic. Some (such as Akamai [36]), use DNS to translate the hostname of a page request into the IP address of an appropriate server. This translation may consider the location of the client, the location of the server, the connectivity of the client to the server, the load on the server, and other performance and cost based criteria.

An example that shows how the CDN infrastructure is embedded in the Internet architecture is shown in Figure 1. The Internet is divided into a collection of autonomous systems (ASs). Each AS is managed by an Internet Service Provider (ISP), who operates a backbone network that provides connectivity to clients and to other ISPs. Figure 1 shows four ASs, numbered 1–4, two Web site publishers, `home.ex` and `adserver.ex`, and two sets of clients. The publisher `home.ex` is connected to AS 3 while the publisher `adserver.ex` is connected to AS 2. A set of clients is connected to AS 1, another to AS 4. Traffic is routed between the ASs by means of Exterior Gateway Protocols [37]; BGP [38] is the de-facto standard. Traffic within an AS is routed by means of Interior Gateway Protocols [37].

The location of the CDN's servers differ from CDN to CDN and depends on contractual agreements between the CDN and the individual ISPs. In some instances, the CDN servers are deployed within the data centers of the ISP and therefore belong to the same AS, like AS 1, 2, 4 in Figure 1. Clients of the ISP (end users) will typically be served by these servers in the same AS. With other ISPs, the CDN may have a private peering agreement that allows the CDN to serve requests from the ISPs clients via a direct connection between the CDN and the AS. The CDN may also co-locate servers with the ISP's clients, e.g., on university campuses. With other ISPs there may be no relationship with the CDN, and the traffic to the ISP's clients is routed via another AS.

Let us consider the steps that are necessary to download the Web page shown in Figure 2. This page consists of one main page located at `home.ex/index.htm` and four embedded objects. The publisher responsible for `home.ex` has decided to use the services of a CDN, `cdn.ex`. One object (`ex2.gif`) of the sample page is located on the same server as the page itself (`index.htm`); another object (`ex3.gif`) is served by a company providing dynamic advertisements, `adserver.ex`; and objects `ex1.gif` and `ex4.jpg` are hosted by the CDN.

If a specific client from client set A in Figure 1 accesses the Web page, publisher `home.ex` will serve the bytes for the main page and one embedded object, publisher `adserver.ex` will serve the bytes for the object located on its servers, and the "nearest" CDN server will serve the two CDN-located objects—in this case, they will be served from AS 1. In contrast, if a specific client from client set B accesses the page, the two CDN objects will be delivered from a different CDN server, namely the one in AS 4. Keep in mind that it is the objective of the CDN to direct the client to a CDN server that is close to the client.

3. INTERDOMAIN WEB TRAFFIC DEMANDS

In this section we motivate and introduce abstractions for publisher demands and Web traffic demands and discuss some possible applications based on these abstractions.

The interplay between content hosting, intra- and interdomain routing, and the Internet architecture affects the set of traffic demands we choose to estimate. In contrast to previous work [11, 12, 13, 39, 14, 40], we are not focusing on a single ISP. Rather the goal of this study is interdomain traffic imposed by any client accessing content provided by many publishers.

The situation naturally lends itself to two abstractions:

1. a *publisher demand matrix* that captures traffic behavior at the aggregate level of a publisher or content provider; it pairs each client IP block with various publishers and
2. a *Web traffic demand matrix* that captures the traffic at the granularity of a Web server with a specific IP address; it pairs each client IP block with various Web server IP addresses.

Motivation: Traffic demands usually specify the amount of traffic flowing between two end-points, from the source to the destination, which is sufficient as long as both end-points are of the same granularity. In the context of Web traffic, treating end-points at the same granularity is problematic, as there are many more clients than servers or publishers. Distinguishing between individual clients is moot due to the sheer size of the resulting matrix.

Just as the interplay between intra- and interdomain routing motivated a point-to-multipoint demand model [11], it motivates us to define Web demands in terms of network prefixes that are consistent with BGP. This enables us to address questions arising in the context of inter- and intra-domain routing as well as questions regarding how to multi-home sites and how to balance traffic between ISPs.

Summarizing clients according to network prefixes appears appropriate. Network prefixes provide a way of aggregating client traffic that preserves locality in terms of the Internet architecture. Such an aggregation is necessary in order to avoid the severe scalability problems of representing each client at the level of an IP address. In addition, it reduces the statistical significance problem caused by too little traffic per individual IP address.

Yet, summarizing publishers via network prefixes is hazardous. A publisher that serves tens to hundreds of megabits/second to clients is likely to use a distributed infrastructure together with some load distribution mechanism, such as DNS round-robin or proximity-aware routing. In general these mechanisms are very similar to those employed by CDNs. This usually means that the content is available via multiple IPs in different network prefixes. Furthermore, it is sometimes impossible to deduce the Web site publisher from its IP address: A server may host multiple sites of several publishers. Even the URL of an object does not directly allow us to infer the identity of the publisher for the content, e.g., that Vivendi Universal Interactive Publishing is responsible for `www.lordoftherings.com`. Some publishers split their content into various sites, each with its own responsible organization and its own independent infrastructure. This implies that one may want to capture the traffic matrix at two levels of abstractions: at the publisher level or at the level of each individual Web server.

Illustrative Examples: Having motivated the need for the two kinds of matrices—publisher demand and Web traffic demand—we now present some illustrative examples. Figure 3 shows two

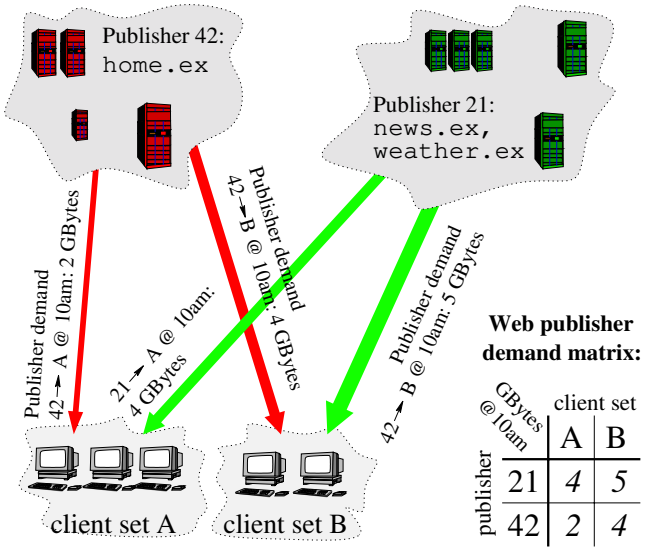


Figure 3: Publisher demands.

different publishers that are identified by id numbers 42 and 21, and the domain names of the sites that they publish: `home.ex` for 42 and `news.ex/weather.ex` for 21. Their content is accessed by two different client sets: A and B. Each client set accesses some of the content provided by `home.ex` and `news.ex/weather.ex`. This results in traffic flowing from the Web sites of `home.ex` and `news.ex/weather.ex` to the client sets A and B. These traffic flows are what we refer to as *publisher demands*.

If we want to improve, say, our routing decisions, then publisher demands are not of much use: They do not take into account the server locations. In the distributed infrastructure for the publisher with ID 42 shown in Figure 1, some of 42's content (namely `home.ex`) is hosted at servers connected directly to AS 3, some of 42's content has been offloaded to a CDN; furthermore there may be third-party content such as banner ads hosted by `adserver.ex` on some of 42's pages. In Figure 1, the resulting three *Web traffic demands* to client set A are indicated by the smooth arrows; the Web traffic demands to client set B are depicted by the dotted arrows.

Applications: These notions of demands enable experimentation with changes to content hosting, to routing, to the AS level topology, as well as to the location of the content and/or the clients. A publisher that needs to upgrade its infrastructure has many choices: upgrade the existing servers, add more servers, add more bandwidth to existing network connections, add alternative network connections, change the way requests are allocated to individual servers, or outsource more of its content delivery. In order to decide on the best option, the publisher may use the publisher demands to evaluate possible scenarios: the traffic volume imposed by different client sets may influence his decisions. For such "what if" scenarios he needs to understand the dynamics of both the publisher demands as well as the Web traffic demands as well as the differences in the dynamics between them.

An ISP may also need to predict the effects that adding or moving a link or peering session may have. This requires a model of interdomain traffic. An important difference between traffic statistics collected within an AS and the Web traffic demands discussed here is that they describe traffic flows, not just through the network of the ISP, but throughout the Internet. Therefore, given an understanding of the dynamics of Web traffic demands, it is easier to estimate the effects that decisions (such as adding peering connections) may have. Furthermore it is possible to explore what effects policy changes will have. For the first time this is now feasible not just for policy changes by the ISP itself but also for policy changes by other ISPs.

By combining Web traffic demands with topology and BGP routing information one can explore the impact of routing instabilities on actual traffic flows and vice versa. Furthermore by combining the Web traffic demands with performance measurements one can explore how user feedback should be factored into future decisions. Furthermore both demands, the Web traffic demand as well as the publisher demand, are ideal inputs for driving interdomain network simulations.

4. USING CDNS TO ESTIMATE PUBLISHER DEMANDS

Computing the publisher demands is possible given either information from each publisher regarding which clients access the content served by that publisher from which prefixes, or given information from each client set about which Web sites they are requesting. One way of deriving this information would be to collect fine-grain traffic measurements at all publisher sites or all client sites. This may enable us to identify the traffic as it reaches the Web site publisher or the clients. However, this approach is virtually impossible since the huge number of publishers/client sets imposes a task that is unmanageable. Furthermore it would still be necessary to address the question of how to distinguish publishers co-located at a server. Just analyzing a large proxy log does not help either, since it does not allow us to gather information about any significant subset of all possible clients.

Instead, we focus on publishers, because there are far fewer publishers than clients. Yet, instead of considering all publishers, we take advantage of the fact that CDNs provide (Section 2.2) reverse proxy services for the content providers (the publishers). They are acting as "subcontractors" to the publishers. Using data collected within CDNs has several advantages:

- CDNs serve the content on behalf of their customers (the publishers). This implies that the CDN has a way of relating content to publishers.
- Due to the requirements imposed by volume-based billing, CDNs collect data on behalf of the publishers regarding how much traffic is served. This implies that the CDN has a way of deducing the amount of traffic it serves on behalf of each individual publisher.
- In addition, most publishers do not want to lose access to the information they can collect when they serve content directly to clients. For example, information about which clients are accessing what content is derivable from Web server logs. Accordingly the CDN has to collect this "Web server"-like log information. As a consequence, it has a way of relating traffic to clients.

Moreover the number of CDN service providers is significantly smaller than the number of publishers. A list of CDN types and their products is maintained by Teeuw [41] and Davison [42]. To further reduce the candidate set, we observe that the market is dominated by only a small number of CDNs such as Akamai, Speedera, Cable & Wireless and Mirror Image.

Focusing on CDNs limits us in terms of the number and kind of publisher demands that can be estimated: If a publisher has no association with a CDN, it will not be possible to derive his publisher demands. This raises the question of which publisher demands we are interested in, and if those are likely to be associated with a CDN. Like a lot of other quantities in networking [43, 44, 11] and elsewhere [45], we expect publisher demands to be consistent with a Zipf-like distribution. A Zipf-like distribution is one where the contribution of the k-th most popular item varies as $1/k^{\alpha}$, for some α. Since the heavy hitters account for a significant part of the traffic, we are mainly interested in them. Luckily those are the ones that are more likely to use the services of a CDN. Therefore CDNs can provide us with a way of estimating the publisher demands for those content providers that are most popular and thus account for a large part of the traffic.

Still one problem remains: as discussed in Section 2.2 and as shown in Figure 1, CDNs try to take advantage of their distributed infrastructure by serving traffic locally. Thus, how can we expect to derive estimates for interdomain Web traffic demands from traffic to CDNs? Here it turns out that most publishers will not serve their whole content via the CDN. Rather they will use some mixture as shown in Figure 2. Note that not all content has to be served via the Web site of the publisher or the CDN; rather some embedded objects may be located on yet another server, e.g., banner advertisements.

Together this provides us with the opportunity that we need. If we know the ratio of a customer's traffic serviced via a CDN vs. via the servers of the publisher vs. via external sites, see Figure 4(a), and if we know the traffic serviced by the CDN, see Figure 4(b), we can estimate the other amounts, see Figure 4(c). These facts allow us to estimate publisher and Web traffic demands for *all* client prefixes world-wide and *all* publishers that are customers of the CDN. Our methodology significantly improves the availability of interdomain traffic estimation—so far at best a scarce quantity.

5. ESTIMATING INTERDOMAIN TRAFFIC DEMANDS: REALIZATION IDEAS

With access to the logs of a CDN, determining the traffic served by a CDN on behalf of a specific publisher is possible. Accordingly we now discuss how we approach the remaining problems: how to estimate traffic ratios between publisher and CDN traffic, as well as how to map publisher demands to Web traffic demands. Further details are provided in the Appendix.

Estimating traffic ratios: One way to proceed is to explore the content provided by the Web site of the publisher offline. Given a set of Web pages one can easily calculate the fractions of data served by the CDN vs. the fraction of data served by the original Web site. The problem with this approach is that it ignores the fact that certain Web pages are more popular than others.

Hence, we really need access to information about user accesses. There are many ways of doing this [46]: from users running modified browsers [5], from the logs of the publishers themselves [7], from proxies logging information about which data is requested by the users of the proxy [47, 48] or from the wire via packet monitoring [49, 50, 51]. Each of these methods has its advantages and most have severe limitations regarding the detail of information that they log. Distributing modified Web browsers suffers from access to the browser software and from users not accepting the modified browsers. While a few publishers might cooperate by revealing their logs, most will not. In addition, this approach suffers from a scalability problem. Using proxy logs or logs derived via packet monitoring is more scalable with regards to ISPs. But with regards to the size of the user population that can be monitored, it is more limited.

To choose the appropriate solution let us consider the granularity at which we need the information. The purpose of estimating the publisher demands is mainly to understand their medium time-scale fluctuations and their impact on traffic engineering, routing, etc. We are not as interested in small time-scale events (and in any case it is hard to understand their causes). Therefore some coarse-grain estimation is sufficient for our purposes. Hence we propose the following two-fold approach:

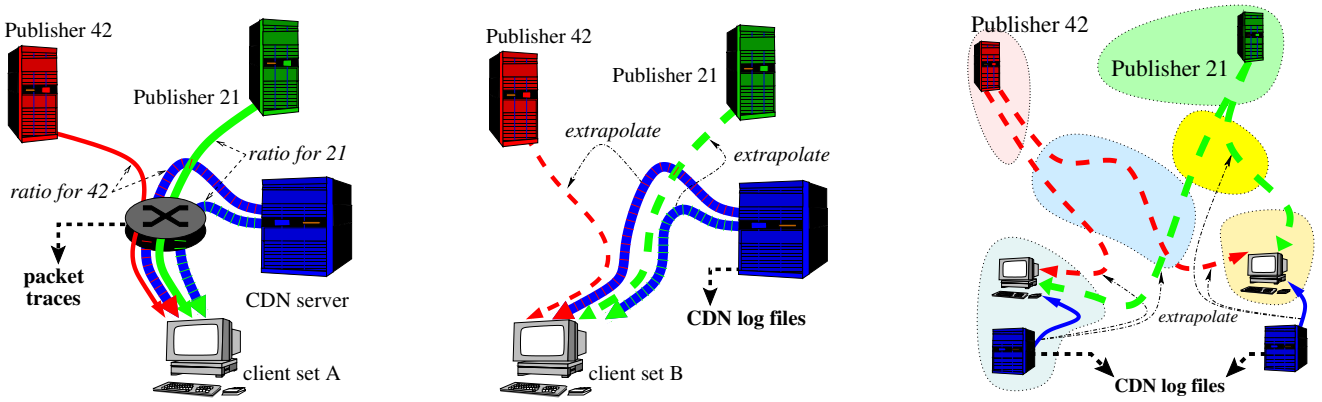
- to obtain from the publisher their estimate of the fraction of traffic that is served by the CDN and other third party providers; admittedly, we utilize the provider-customer relationship between the CDN and the publisher to acquire this information, which is provided by only a subset of the publishers.
- to use packet-level traces or proxy logs to derive the fractions for some users and therefore for some sample client sets. (While proxy logs suffice, since detailed timing information is not required, the analysis in this paper is based on packet-level traces.)

Figure 2 shows an example of a Web page. A log file, derived from a proxy log or the packet traces, should show six entries per access to this page, i.e., one for each object (unless it is cached in the user's cache). Each entry includes an `object_id` (i.e., the URL), the start and end time of the download of the object, the transferred bytes, and the `HTTP_REFERER` field (if specified by the user agent). Note that the referrer field, which lets a user agent include the URL of the resource from which the requested object was reached, is optional and not necessary. Nevertheless most popular Web clients, such as Internet Explorer and Netscape, include them regularly. They prove to be extremely helpful. In our sample page, all embedded objects have the same value for their referrer field independent of where the object actually resides. Indeed the value is the same as the URL of the base page. Thus the referrer field provides us with the means to associate the objects and therefore provide us with the means of estimating the ratios between the traffic flows.

One way of estimating the ratios would be to try to compute the exact temporal and causal relationship between the pages and their embedded objects. But past work, e.g., in the context of estimating the benefits of prefetching [48] or piggybacked cache validation [46], has shown that this is a nontrivial task, especially in the presence of proxies and strange users. For our purpose the fact that there *is* a relationship is sufficient. See Appendix B for details.

From publisher demands to Web traffic demands: In order to derive the Web traffic demands from the publisher demands, we first need to map the Web sites of the publishers to IP addresses. This mapping may not be a one-to-one mapping. Recall that some publishers use a distributed infrastructure and therefore apply DNS mechanisms for "load balancing", "proximity-aware", or "server-feedback dependent" name resolution, in a manner similar to Akamai's mechanism for distributing load, or even entrusting Akamai to provide these mechanisms.

Again, we propose to take advantage of information available to the CDN. It knows the set of hostnames that is associated with



(a) Determining traffic ratio publisher traffic/CDN traffic

(b) Estimating unseen non-CDN traffic

(c) Resulting interdomain traffic demand matrix

Figure 4: Web publisher demand estimation.

each publisher. Therefore the problem is reduced to associating each hostname with its set of IP addresses.

This can be done using DNS queries. To account for "proximity-aware" or "server-feedback dependent" policies used by the publisher, it is not sufficient to issue DNS queries from a single point in the Internet—rather we need to use a set of DNS servers that are distributed throughout the Internet. Since we have to issue recursive queries[1] to these servers in order to discover their view of the server IP addresses, they have to allow recursive DNS queries.

In a second step, we determine which server is used by which client. This problem can either be extremely simple or extremely hard. If the site uses a single IP address or simple DNS round robin across a number of different IP addresses, this step is trivial. Since DNS round robin is supposed to partition the requests about evenly across all of the servers, this is what we will do in estimating demand. If the site uses a more sophisticated mechanism, we are left with a fairly difficult problem. Here we have two possible ways to approximate the decision of the physical Web site: We can either use the result of the DNS server "closest" to the client set, or we can assume that the client set is directed to the "closest" server. Here we propose to capture the meaning of "close" in terms of AS distance. This seems reasonable, since other measures of closeness are even harder to define, and since it is known that some distributed infrastructures are using this information [52].

More details concerning our implementation are provided in Appendix C.

6. DATA SETS

The computation of the demands draws on several different data sets, as summarized in Figure 5 and 6. This section describes our approach for harvesting and preparing these various large data sets, each collected at a different location at a different granularity.

From the CDN: Using logs that feed into the CDN billing system of a major CDN provider, Akamai, we extract for each client set how much content from which publisher is accessed (after appropriate anonymization). Each individual log file records all accesses to some part of the CDN infrastructure during some time period and is available for processing some time after the last recorded access.

[1]In an iterative query, the contacted name server tells the requesting name server which name server to ask next, while in a recursive query the contacted name server proceeds by sending a query to the next name server on behalf of the original user.

We captured logs for three two-hour time periods: 9–11:00 h UTC on Mon Apr. 26th, 2004 (CDN1) and 8:30–10:30 h UTC on Wed Apr. 28th, 2004 (CDN2) and 17–19 h UTC on Wed May 5th, 2004 (CDN3) from more than 90 / 85 / 65% of all the operational servers of the CDN [2]. There are two reasons why we did not capture logs from all servers: Logs for certain time periods arrive in bursts imposing a huge instantaneous burst overloading our limited research collection infrastructure. Other logs can be delayed due to remote network outages, and even arrive after we stopped our data collection process. In addition the online collection is augmented by an offline retrieval of some subset of the logs via an archival system. We initially aggregated this data using the methodology described in Figure 19 using a time aggregation of half an hour. This time aggregation was chosen to examine the spatial rather than the temporal variability of the data.

From three user sets: Three sets of client access information were extracted from packet-level traces at the 1 Gbit/s upstream link of the Münchner Wissenschaftsnetz (*MWN*) in Germany. The *MWN* provides external Internet connectivity to two major universities (Ludwig-Maximilians-Universität München, Technische Universität München) and a number of smaller universities, government organizations, and research institutes. Overall the network contains about 50,000 individual hosts and 65,000 registered users. On a typical day the *MWN* exchanges 1–2 TB of data with its upstream provider. On the 13th of May during the day (8–20 h), 295.5 GB used the HTTP port, which corresponds to 26.5% of the traffic. During the night 112.2 GB (18%) of the traffic was HTTP. This indicates that the Web is still a major traffic contributor.

Monitoring is realized via a monitoring port on a Gigabit Ethernet switch just before the traffic passes the last router to the Internet. We captured the raw packet stream using `tcpdump` on disk and then extracted the HTTP connections offline using the HTTP analyzer of the intrusion detection system `bro` [53]. The resulting trace contains all relevant HTTP header information and is much more compact than the raw packet data.

Since extracting HTTP data at Gigabit speed is almost impossible using standard PC hardware [49] we split our client base into three groups: one for each university (TUM, LMU) and one that covers the other organizations (MISC). To ensure a reasonable coverage of all client groups, we monitored each client group for a

[2]The relatively bad coverage for the May dataset is due to having to use a compute server for retrieving and storing the logs.

Dataset	Obtained from	Key Fields
CDN sites	CDN	List of Web sites and Web site publishers that use the CDN
CDN servers	CDN	List of hostnames of Web sites
CDN logs	CDN billing system	Per accessed object: client IP address, resource, start and end time, transferred bytes
HTTP logs	external network connection	Per accessed object: user IP address, url, start and end time, transfered bytes, referrer, hostname
DNS lookups	set of name servers	Per hostname and DNS server: set of IP addresses
BGP table	peering points	Per network: set of possible routes (AS-path)

Figure 5: Datasets and key fields used in computing and validating the publisher and content traffic demands.

Dataset	Date	Duration	Size
CDN logs	{04/26,04/28,05/05}/04	3×2 hrs	617.4 GB `.gz`
HTTP logs	01/30/04–05/11/04	102 days	28.5 GB `.gz`
DNS lookups	5/12/04–5/13/04	1 day	5.4M queries
BGP tables	4/28/04	—	270 tables

Figure 6: Per data set summary information.

2-hour period, rotating through the groups. Accordingly each trace captures all downloads of all clients in the group from all publishers as well as the CDN. In total, we collected 1,017 traces, each of which covers a 2-hour period. This approach ensures reasonable packet loss rates. Of the 1,017 measurement intervals, the number of intervals with more than 0.1% / 1% / 10% packet drops (as reported by `tcpdump`) was 124 / 22 / 1. The maximum packet loss rate was 10.18%, the average is 0.23%, and the median is 0.0028%.

From the DNS system: We identified roughly 7,000 DNS servers using a different packet level trace, while ensuring that each server supports recursive queries. But the process does not pay attention to the distribution of the DNS servers within the Internet infrastructure. Therefore in the next step we identified a subset of 516 DNS servers that return different results when resolving the name of the main CDN Web server. The 516 DNS servers are located in 437 ASs in over 60 countries. We restrict ourself to using this subset in order to reduce the load on the overall DNS system while achieving a good coverage of the Internet infrastructure. To resolve which publishers are using a distributed infrastructure, we selected a subset of 12,901 hostnames used by the publishers. The resolution of these hostnames resulted in more than 5.4 million queries of which 98.2% received a valid response.

From the Routing system: We constructed a joined BGP routing table from the individual BGP tables on the 4/28/04 from RouteView [54] and RIPE's RIS project [55]. This table contains 161,991 routable entries. Furthermore we extracted an approximation of the contractual relationships between the AS using a methodology similar to that proposed by Gao [56].

7. EXPERIMENTAL RESULTS

In this section, we present our initial results of applying our methodology to the various data sets discussed in Section 6.

7.1 Estimating CDN publisher demands

The first step is estimating how much traffic is sent by the CDN on behalf of each publisher to each client set. For the initial analysis in this paper, we decided to use static groups of /24 prefixes to define client sets. We observe 1,130,353 different client sets within the datasets CDN1 and CDN2. This corresponds to a 23.6% coverage of the overall IPv4 address space and 52% coverage of prefixes within the routable IPv4 address space. 1.3% of the observed client space is not publicly routable, perhaps due to placement of CDN servers within private networks. In total the client sets accessed roughly 41 Terabytes of data via the CDN network. Thus on average, each client set accessed about 36 MBytes over the three trace periods.

The Internet has obviously many client sets and a sizable number of publishers. But who is contributing the majority of the traffic—is it a small set of client sets, or a small subset of the publishers? Even by just studying the amount of traffic serviced by the CDN, we can get a first impression of these relationships. In Figure 7, we rank client sets by total traffic received from the CDN from largest to smallest, and plot the percentage of the total traffic attributable to each for each 30 minute time interval of the CDN2 trace. This corresponds to plotting the (empirical) complementary cumulative distribution function (CCDF) of the traffic volume per client set. In order to not obscure the details in the curves we use lines instead of marking each point for ranks greater than five. To better distinguish the curves we add some supporting markers. As predicted, we find a "linear" relationship on the log-log scale, an indication that the distribution is consistent with the characteristics of a Zipf-like distribution [45, 43]. The client sets are sorted by their activity in terms of downloaded bytes; the first client set is the most active one. This implies that one has to look for the linear relationship in the left part of the plot, while artifacts can be expected at the right side.

But do client sets exhibit the same sort of activity distribution even if we focus on individual publishers rather than on all publishers taken together? In Figure 8, we explore the characteristics of the top 10 publishers, selected by the total number of bytes that they serve to all client sets (using the same plotting technique as before). The fact that we still observe a "linear" relationship on the log-log scale indicates that even single publisher demands are dominated by the behavior of a few client sets. One aspect that may be contributing to these effects is that client sets are located in different time zones. About 40.4% of the client sets in CDN1 and CDN2 are located in the US, 9.4% in Japan, 6.0% in Korea, 4.2% in the UK, 4.2% in China, 3.9% in Germany. (The mapping of network to country is done via Akamai's EdgeScape tool.) One reason for reduced demands is that for some client groups most are sleeping while users of other client sets are at work, etc. While the impact of time zones has to be further explored, we start by subselecting various subsets of client sets. Each of these client sets covers either one (Japan), two (UK, France, Germany), or four time zones (US). We still observe activity drops that are consistent with Zipf-like distributions (plots not shown) if we split the demands per client or per time. The bends for Publishers 6 and 10 in Figure 8 are due to the superpositions of access by client sets in the US and abroad. The ones in the US have a higher demand than those outside the US.

Even though the client sets in Figure 8 are ranked separately, according to their activity for each publisher, it also shows that a client set that receives the most bytes from one publisher does not do so from another publisher. Rather, there are significant differences. This indicates that each publisher in the Internet has to

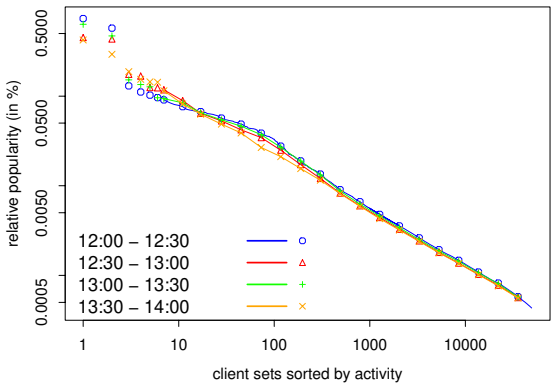


Figure 7: CCDF of client set traffic volume (% bytes served from all publishers each 30 min).

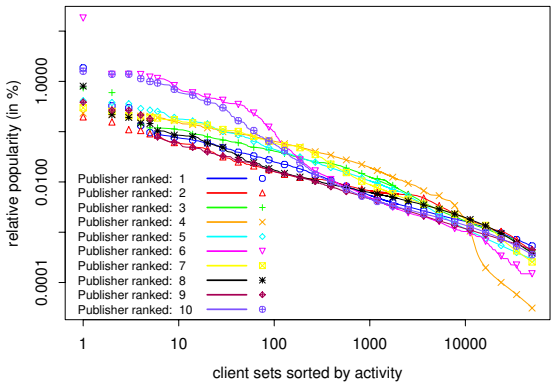


Figure 8: CCDF of client set traffic volume (% bytes served) per top-10 publisher during the two hour period of CDN2.

determine for itself who the heavy hitters (contributors) among the clients are—extrapolating from one client set to another can be misleading.

But what is the behavior if we consider the data from the viewpoint of the client sets? In Figure 9 we explore the popularity of content served by the CDN on behalf the publishers (using the same plotting technique as before). Again we observe a curve that indicates a Zipf-like distribution in the range of 1–1,000. The dropoff in the curve for less popular publishers indicates that there is a large

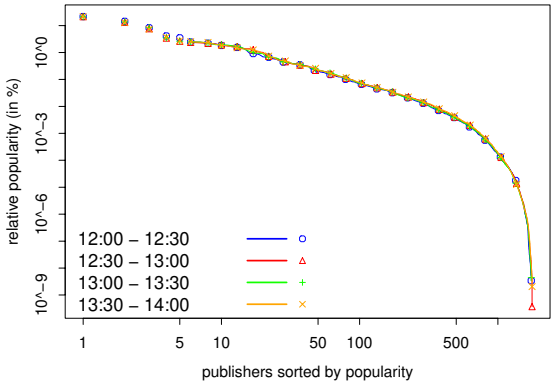


Figure 9: CCDF of publisher traffic volume (% bytes served to all client sets each 30 min).

number of publishers that do not serve a lot of data via the CDN. This does not disprove that, for the popular publishers, the distribution is consistent with a Zipf-like distribution.

Generally, we observe the same kind of curves for all data sets and for each subset of the datasets. For example, in Figure 9, the curves for the publisher popularity in terms of traffic volume between consecutive 30-minute time periods fall on top of each other. The same observations hold if we look at the individual publishers or the client sets over consecutive 30-minute intervals. But this does not imply that its always the same publisher or the same client set that dominates the distribution. Accordingly Figure 10 plots the bytes contributed by each country during one 30-minute time period vs. another 30-minute time period. The left plot does so for consecutive time periods. The nice concentration around the diagonal indicates that the volume changes are not rapid within any of the three datasets. In contrast, the right plot shows the same kind of plot comparing corresponding 30-minute time periods from the 26th of April to those of the 5th of May. (A 30-minute time period starting at offset x in one trace corresponds to the 30-minute time period starting at offset x within the other trace.) Note that, due to the time shift, one should expect a larger spread. This is indeed the case, indicating that the popularity changes have to be considered not being just time-of-day variations.

7.2 Estimating relationships between CDN and publisher flows

Once we know how much Web traffic is flowing from the CDN to each client set, we need the ratios to extrapolate from the partial CDN publisher demands to the Web publisher demands. Accordingly we apply the our methodology to the client access logs. (Further details are provided in Figure 20 in the Appendix.) Note that we are not necessarily capturing all of the traffic from the publisher since our methodology is based on the referrer fields in the requests for CDN-delivered objects, i.e., there might be even more CDN customer data being delivered than we are estimating.

We start with presenting some basic characteristics of the data sets from the three client populations covering all monitored subnets, see Figure 11. Overall, in the TUM, LMU, and MISC data sets, we observed roughly 522 million different requests for Web objects for more than 5.15 TBytes of data. This implies that the mean object size in our data sets is about 9.5 KBytes. The mean size of an object served by the CDN to the clients at TUM, LMU, and MISC is a bit smaller at about 8 KBytes. This accounts for the difference between the % requests directed towards the CDN vs. the % of bytes. While 4.2–4.9% of all HTTP requests are served by the CDN, this corresponds to only 3.14–4.31% of the HTTP bytes.

From Figure 11, we see that the clients only retrieve 1.8–2.2% of the HTTP bytes from the CDN customers themselves. This indicates that the ratio of bytes served by the CDN vs. the bytes served by the publishers can vary from 1.4 to 2.5: The relative percentage of requests directed to the CDN customers is larger than the relative percentage of bytes retrieved from the CDN. This indicates that CDN customers delegate their larger content to the CDN, which is to be expected. Yet while publishers delegate a large amount, they do not delegate all of their traffic. Therefore our approach for estimating publisher traffic can be expected to yield estimates of interesting interdomain traffic flows for a significant fraction of the overall traffic volume.

The fraction of bytes in the category *related to non-CDN-customers* gives us another possible avenue for estimating interdomain

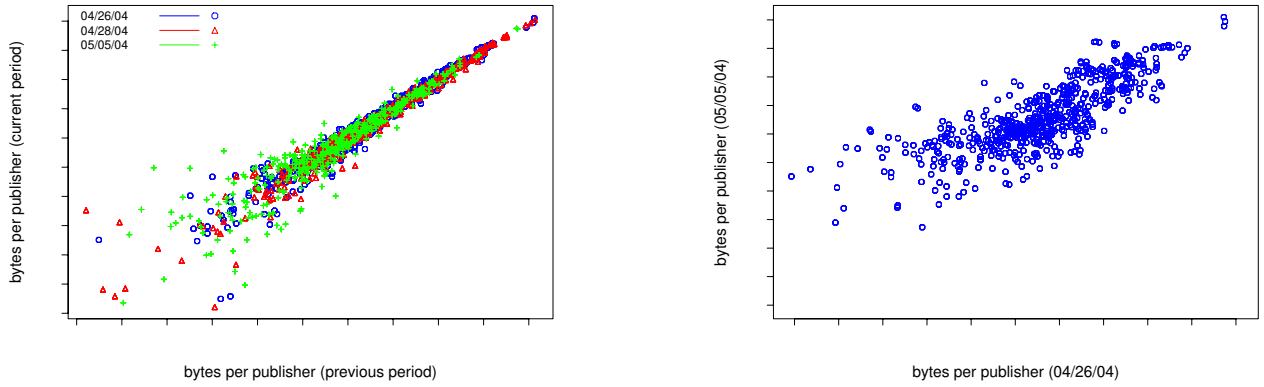


Figure 10: Scatterplot: publisher bytes for time period t vs. period t'.

Users	Description	Requests (in K)		Bytes (in Gbytes)	
		absolute	relative	absolute	relative
TUM	Total	357,621	100.00%	3795.83	100.00%
LMU	Total	91,104	100.00%	721.60	100.00%
MISC	Total	62,013	100.00%	636.47	100.00%
All	Total	510,738	3 × 100%	5153.90	3 × 100%
TUM	CDN	15,065	4.21%	119.00	3.14%
LMU	CDN	4,449	4.88%	26.75	3.71%
MISC	CDN	3,043	4.91%	27.40	4.31%
TUM	CDN customer	10,650	2.98%	83.95	2.21%
LMU	CDN customer	2,549	2.87%	13.75	1.91%
MISC	CDN customer	2,107	3.40%	11.20	1.76%
TUM	related non-CDN	6,121	1.71%	44.61	1.18%
LMU	related non-CDN	1,325	1.45%	5.15	0.71%
MISC	related non-CDN	1,212	1.76%	4.91	0.77%

Figure 11: Basic statistics of the user access characteristics.

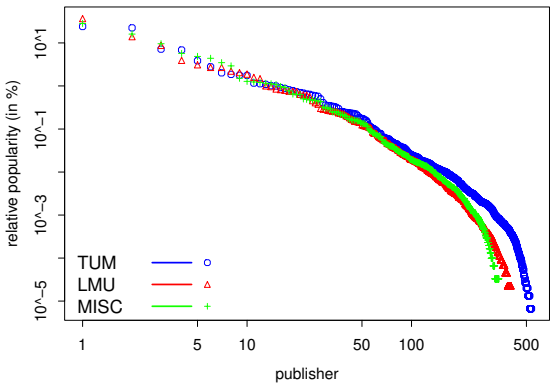


Figure 12: CCDF of requests per publisher.

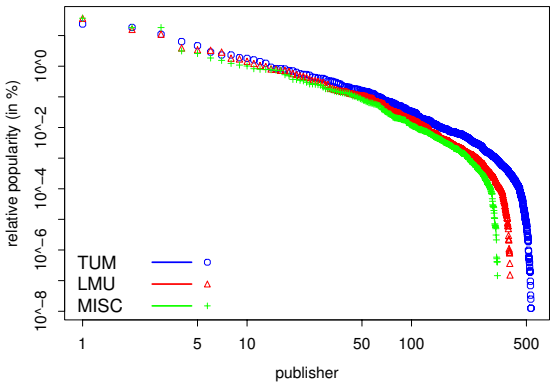


Figure 13: CCDF of bytes per publisher.

traffic flows. There are two reasons why requests/traffic falls into this category: publishers offload some of the content to other service providers (e.g., those providing targeted advertisement), and some of the publisher's content is served in connection with other sites (e.g., advertisements on someone else's Web page). While this indicates some additional potential, in this initial exploration phase we focus on the ratio of traffic served by the CDN on behalf of a publisher vs. the traffic to the publisher itself.

For this purpose we need to associate the bytes served by the CDN and the bytes served by CDN customers' own servers with the appropriate publisher. Using Akamai-internal information sources, we were able to identify 23 million requests from the *MWN* to Akamai-hosted URLs (Figure 5). While 23 million requests are a sizable number, the individual number of requests for objects served by the CDN over smaller time period (2 hrs) are significantly smaller. Averaged over the whole duration of the trace collection this implies that one can expect to see only 2,000–20,000 requests in each data set for each two hour time period. Of course just averaging is unfair since there will be many more requests during busy hours than during off-hours, e.g., in the middle of the night. In addition some subnets, e.g., those with Web proxies, generated many more requests than others. Nevertheless it points out the problem of observing enough samples for deriving a reasonable ratio estimate.

Here we receive help from a trend that has been observed in many other contexts: some publishers have much more popular content than others. We rank the number of requests (Figure 12) and bytes (Figure 13) by provider from the largest to smallest for both data sets, and plot the percentage of total requests/bytes attributed to each. For those publishers that contribute a significant percentage of the bytes these curves are "linear" on a log-log scale. Again this characteristic is consistent with a Zipf-like distribution. Together these two observations imply that we can expect to find time periods with a reasonable number of observations for some significant subset of the publishers in our user access data sets. We now focus on those (*time period*,*publisher*) pairs with enough observations.

Here we define "enough" as observing at least 50,000 requests satisfied by the CDN on behalf of a publisher and 500 requests served by each publisher itself per aggregation period. Using a value of 500 is fairly arbitrary; further investigation is needed to

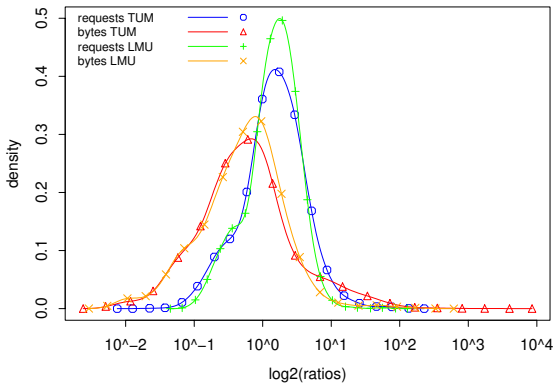


Figure 14: Density of log_2 of ratios for objects of Akamai customers requested from TUM and LMU. Each ratio is calculated via $\left(\frac{\#\textit{requests or bytes Akamai servers}}{\#\textit{requests or bytes publisher-owned servers}}\right)$.

provide a sound basis for a good cutoff value. Using these selection criteria, we compute the ratios of bytes for each publisher and each aggregation period. Not too surprisingly we found that the ratios span quite a wide range of values: from 0.01 to 100. Comparing ratios is awkward, e.g., is the "difference" between 0.03 and 0.06 the same as between 16 and 32? In this context the answer is yes since both "differ" by a factor of 2. Therefore, to ease comparisons of ratios we, in all further discussion, use the binary logarithm of the ratios. Accordingly 0.03 is transformed to -5, 0.06 to -4, 16 to 4, and 32 to 5. Now the differences in both cases are 1. Figure 14 plots the density of the transformed ratios for the TUM and LMU data sets for both bytes as well as requests. We observe for all data sets that the ratios span a significant range of values from -10 to 10 both for requests as well as for bytes. This indicates that different providers use different policies with regards to delegating their information to the CDN. Furthermore we see, as expected, that the CDN usually provides more bytes than the original publisher for most but not all publishers. In addition with regards to requests the distribution is more balanced. This indicates that some publishers use the CDN for big objects, such as software distribution.

While the overall distribution of the ratios is interesting, more relevant for the purpose of estimating the publisher demands is the question: How stable are the ratios across time and user populations? Overall it is well known that traffic volume [6] and flow arrival streams [57] are self-similar and exhibit significant burstiness. Therefore we can expect some fluctuations with regards to the number of requests over time. In addition, not every user will access the same pages from the publisher, and different subsets of pages will lead to different ratios in terms of bytes from the publisher and the CDN. But what are the impacts of all these causes of instability? Our estimation methodology allows us to explore the size of these instabilities since it will yield multiple samples of estimated ratio values for various publishers. Figure 15 shows boxplots of the ratios for the 15 most popular publishers for the samples of the three data sets, TUM, LMU, and MISC. Boxplots can be used to display the location, the spread and the skewness of several data sets in one plot: the box shows the limits of the middle half of the data; the line inside the box represents the median; the box widths are proportional to the square root of the number of samples for the box; whiskers are drawn to the nearest value not beyond a standard span from the quantiles; points beyond (outliers) are drawn individually.

Most of the boxes have a reasonably small spread (less than two). But others have quite a spread, e.g., index 4. This is partially due to a fairly small sample size and partially due to the variability of different content that is offered by that publisher. Further aggregation and combining the information from different user sets can sometimes be helpful—Figure 15 also shows the boxplots for the samples from the combined data sets. While some estimations of the ratios stabilize, as indicated by the smaller range of the box, others expand due to the differences in the user behavior.

Generally, we can estimate the ratio of publisher demand serviced by the CDN vs. that serviced by the publisher. But there are drawbacks to this approach: A large number of requests needs to be monitored in order to derive reliable estimations. The estimations can vary across time and some attention has to be paid towards different subject/interest areas by different user sets. Furthermore not all user sets will access sufficiently many objects from all publishers that are customers of the CDN. Therefore this approach should be combined with other approaches for estimating the ratios, e.g., static exploration of the Web site and information from the publisher itself.

7.3 Mapping of publisher demand to Web traffic demands

The next step is to apply our methodology for mapping the publisher demands to Web traffic demands. (Further details are provided in Figure 21 in the Appendix.) The open question is: how well does the proposed methodology of mapping each client set and each hostname to a single server IP address work? This is a two-step process. First, we need to identify the set of IP addresses for each hostname. Then we need to identify which subset of the IP addresses to choose for each client set.

If a hostname is hosted by the CDN itself or if the infrastructure is using DNS round robin by itself, the latter step is simple. In the first case we know which IP address serves the traffic and in the second case all returned IP addresses are used. Using the data described in Section 6 we observe that of the 12,901 hostnames, 2,106 (16.3%) are hosted by the CDN itself, 1,242 (9.6%) are using some form of proximity-aware load balancing, while 10,906 (84.5%) are consistently returning the same set of IP addresses. Of these hostnames, 9,124 (83.8%) are returning a single IP address while 1,079 (8.4%) are utilizing only DNS round robin. Most of these (830) are using two IP addresses, while 79 are using more than five IP addresses. Therefore we have solved the problem for 90.4% of the considered hostnames. If most publishers are serving their content out of a small number of servers, then most clients must be far away from those servers, which indicates that a significant fraction of the traffic that we capture will be interdomain traffic.

This leaves us the with 1,239 hostnames hosted on a distributed infrastructure and using proximity-aware load balancing. To better understand this infrastructure, we show a histogram of the number of IP addresses (Figure 16) and the number of ASs (Figure 16). We observe that most of these hostnames (83.5%) are only mapped to a small number of IP addresses (≤ 5). Indeed more than 34.7% are using only two distinct IP addresses. Next we examine if the infrastructure crosses domains, see Figure 17. 377 (30.4%) of all hostnames using proximity routing are in a single AS. This means that from the view point of interdomain routing, we will not be able to distinguish these demands. We observe that 44% of the hostnames are located in at least two but at most five different ASs.

To explore how the infrastructure of the remaining 862 hostnames is embedded in the Internet we studied the minimal AS distances of the ASs of the IP addresses of the distributed infrastruc-

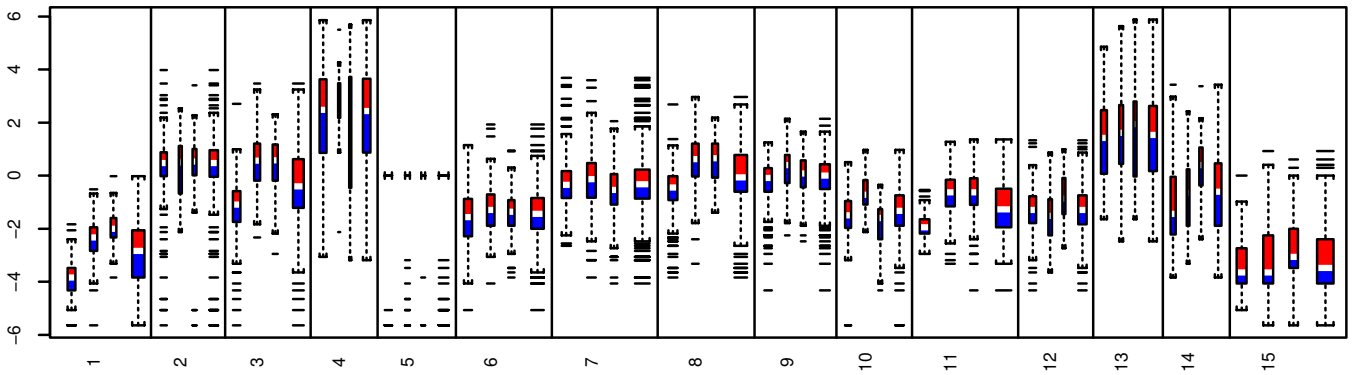


Figure 15: Boxplot of log_2(ratios) for the 15 most popular publishers (labeled 1, 2, ..., 15). For each publisher results for four data sets are shown (from left to right): TUM / LMU / MISC / (TUM ∪ LMU ∪ MISC).

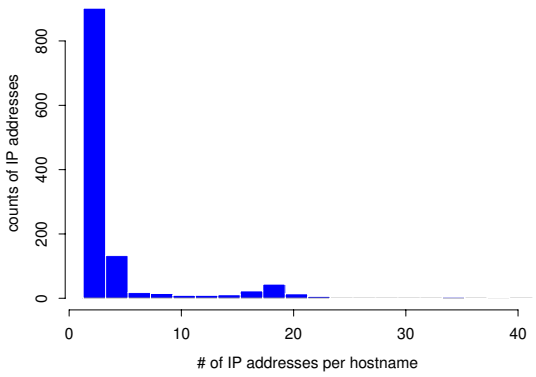


Figure 16: Distributed infrastructures: IP addresses per hostname.

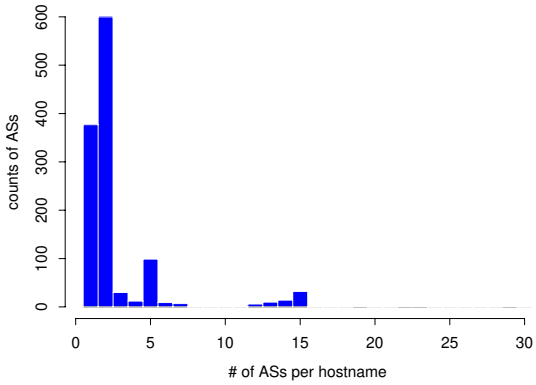


Figure 17: Distributed infrastructures: ASs per hostname.

ture to the ASs of 500 randomly selected IP client sets. In order to compute the distances we consider the contractual relationships as derived from the routing tables [56]. Each AS path may only cross a single peering/sibling edge, and may never follow a customer-to-provider edge once it has followed a provider-to-peer edge. Any edge unclassified by the heuristic is treated as a "sibling/peer" link. We observe, Figure 18, that providers that use more servers and distribute them in various AS indeed gain some benefits. The mean distance and the standard deviation to other ASs is reduced.

8. SUMMARY AND OPEN QUESTIONS

In this paper, we propose two models for interdomain traffic demands, publisher demands and Web traffic demands, that capture

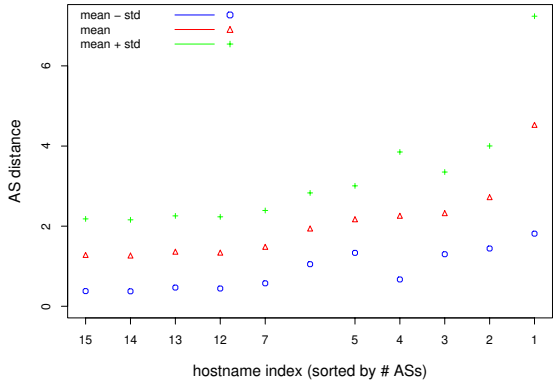


Figure 18: Distributed infrastructures: AS distance between client sets and publisher hostnames.

the origin, the volume, and the destination of the data, and thus provide an interdomain traffic matrix for Web traffic. We believe that this simple abstraction can facilitate a wide range of engineering applications, ranging from traffic engineering, to planning of content delivery, to network simulation. We further present a methodology for populating parts of the demand model using logs from CDN networks, observations from user sets, the DNS, and the routing system.

The experimental results obtained by applying our methodology to logs from a major CDN and two large user sets are promising. Our approach seems to allow us to capture a significant fraction of all Web traffic. Viewed on any scale, but particularly in terms of the number of pairs, our matrices are some of the largest ever generated. We have demonstrated that it is indeed possible to combine server log data from a CDN with packet level traces from large user sets to estimate a good chunk of all interdomain Web traffic as proven by the diversity and coverage of the demands. Nevertheless our results (especially the numerical estimates) should be treated as preliminary and viewed mainly as an indication of the potential of the methodology.

We present a collection of directions for further research:

1. We have captured only one class of traffic, namely HTTP. While several studies have shown that HTTP traffic is among the most common, its dominance has recently been challenged by new classes of traffic such as peer-to-peer file sharing data and streaming media. How well does HTTP traffic demand effectively represent overall traffic demand? How can traffic demand for other classes be estimated?

2. In this work we assume that the number of bytes served by the content provider for each Akamai-served object can be estimated by examining traces from a small number of large client sets. Is the observed ratio of bytes served by the customer to bytes served by the CDN (reasonably) invariant across diverse user sets? At this point we have examined only two. It is possible that content providers might tailor their web pages for different client sets; e.g., a U.S.-based site might choose to serve more compact (fewer bytes) web pages to overseas clients.
3. Now that we have a means of estimating interdomain traffic demands, we are beginning to explore aspects such as temporal (time-of-day) and spatial distributions and analyses of publisher/user dynamics. But we expect it to be even more fruitful to combine this data with routing information, specifically BGP tables. How does BGP respond to network bottlenecks? How do the demands shift in response to routing changes?

Acknowledgments

We are thankful for support from Akamai and *MWN* staff, especially Steve Hill, Eric Olson, and Arthur Berger, for helping us access and understand data used in this work. We are grateful to Robin Sommer for help with the Bro configuration and Arne Wichmann for providing us with the AS topology. Finally, we thank the anonymous reviewers for valuable suggestions regarding the presentation of the material.

9. REFERENCES

[1] M. Roughan, M. Thorup, and Y. Zhang, "Traffic engineering with estimated traffic matrices," in *Proc. ACM Measurement Conference*, 2003.

[2] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, "An information-theoretic approach to traffic matrix estimation," in *Proc. ACM SIGCOMM*, 2003.

[3] S. Saroiu, K. Gummadi, R. Dunn, S. Gribble, and H. Levy, "An analysis of Internet content delivery systems," in *Proc. OSDI*, 2002.

[4] B. Krishnamurthy and J. Rexford, *Web Protocols and Practice*. Addison-Wesley, 2001.

[5] M. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: Evidence and possible causes," *IEEE/ACM Trans. Networking*, 1997.

[6] W. Willinger, V. Paxson, and M. Taqqu, "Self-similarity and Heavy Tails: Structural Modeling of Network Traffic," *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*, 1998.

[7] M. Arlitt and C. Williamson, "Internet Web servers: Workload characterization and implications," *IEEE/ACM Trans. Networking*, 1997.

[8] A. Iyengar, M. Squillante, and L. Zhang, "Analysis and characterization of large-scale Web server access patterns and performance," *World Wide Web*, 1999.

[9] C. Wills and M. Mikhailov, "Studying the impact of more complete server information on Web caching," in *Proc. of the 5th International Web Caching and Content Delivery Workshop*, 2000.

[10] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of Internet traffic engineering." Request for Comments 3272, 2002.

[11] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, J. Rexford, and F. True, "Deriving traffic demands for operational IP networks: Methodology and experience," in *Proc. ACM SIGCOMM*, 2000.

[12] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot, "Traffic matrix estimation: Existing techniques and new directions," in *Proc. ACM SIGCOMM*, 2002.

[13] A. Medina, C. Fraleigh, N. Taft, S. Bhattacharyya, and C. Diot, "A taxonomy of IP traffic matrices," in *Workshop on Scalability and Traffic Control in IP Networks at the SPIE ITCOM+OPTICOMM Conference*, 2002.

[14] X. Xiao, A. Hannan, B. Bailey, and L. Ni, "Traffic engineering with MPLS in the Internet," *IEEE Network Magazine*, 2000.

[15] K. Thompson, G. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," *IEEE Network Magazine*, 1997.

[16] V. Paxson, G. Almes, J. Mahdavi, and M. Mathis, "Framework for IP performance metrics." Request for Comments 2330, 1998.

[17] M. Roughan, A. Greenberg, C. Kalmanek, M. Rumsewicz, J. Yates, and Y. Zhang, "Experience in measuring backbone traffic variability: Models, metrics, measurements and meaning," in *Proc. ACM Measurement Workshop*, 2001.

[18] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast accurate computation of large-scale IP traffic matrices from link loads," in *Proc. ACM SIGMETRICS*, 2003.

[19] A. Soule, A. Nucci, E. Leonardi, R. Cruz, and N. Taft, "How to identify and estimate the largest traffic matrix elements in a dynamic environment," in *Proc. ACM SIGMETRICS*, 2004.

[20] G. Liang and B. Yu, "Pseudo likelihood estimation in network tomography," in *Proc. IEEE INFOCOM*, March 2003.

[21] B. Fortz, J. Rexford, and M. Thorup, "Traffic engineering with traditional IP routing protocols," in *IEEE Communication Magazine*, 2002.

[22] N. Feamster, J. Borkenhagen, and J. Rexford, "Guidelines for interdomain traffic engineering," in *Proc. ACM SIGCOMM*, 2003.

[23] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure, "Interdomain traffic engineering with BGP," *IEEE Communications Magazine, Internet Technology Series*, 2003.

[24] B. Quoitin, S. Uhlig, and O. Bonaventure, "Using redistribution communities for interdomain traffic engineering," in *Quality of Future Internet Services (QoFIS 2002)*, 2002.

[25] S. Uhlig, O. Bonaventure, and B. Quoitin, "Interdomain traffic engineering with minimal BGP configurations," in *18th International Teletraffic Congress (ITC)*, September 2003.

[26] S. Agarwal, C.-N. Chuah, and R. Katz, "OPCA: Robust interdomain policy routing and traffic control," in *IEEE Openarch*, 2003.

[27] J. Winick, S. Jamin, and J. Rexford, "Traffic engineering between neighboring domains," 2002. `http://www.research.att.com/~jrex/papers/interAS.pdf`.

[28] L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker, "On selfish routing in Internet-like environments," in *Proc. ACM SIGCOMM*, August 2003.

[29] B. Lavoie and H. Nielsen, "Web characterization terminology & definitions sheet." `http://www.w3c.org/1999/05/WCA-terms/`.

[30] S. Hull, *Content Delivery Networks: Web Switching for Security, Availability, and Speed*. McGraw-Hill, 2002.

[31] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl, "Globally distributed content delivery," *IEEE Internet Computing*, 2002.

[32] S. Gadde, J. Chase, and M. Rabinovich, "Web caching and content distribution: a view from the interior," *Computer Communications*, 2001.

[33] L. Bent and G. Voelker, "Whole page performance," in *In Proc. of the 7th Int. Workshop on Web Content Caching and Distribution*, 2002.

[34] K. Johnson, J. Carr, M. Day, and M. Kaashoek, "The measured performance of content distribution networks," in *Proceedings of the 5th International Web Caching and Content Delivery Workshop*, 2000.

[35] B. Krishnamurthy, C. Wills, and Y. Zhang, "On the use and performance of content distribution networks," in *Proc. ACM Measurement Workshop*, 2001.

[36] `http://www.akamai.com`.

[37] B. Halabi, *Internet Routing Architectures*. Cisco Press, 1997.

[38] J. Stewart, *BGP4: Inter-Domain Routing in the Internet*. Addison-Wesley, 1999.

[39] C. Barakat, P. Thiran, G. Iannaccone, and C. Diot, "On Internet backbone traffic modeling," in *Proc. ACM SIGMETRICS*, 2002.

[40] N. Duffield and M. Grossglauser, "Trajectory sampling for direct traffic observation," in *Proc. ACM SIGCOMM*, pp. 271–282, 2000.

[41] W. Teeuw, "The CDN state of the art.," 2001. `http://www.trc.nl/Middleware/cdn/ENindex.htm`.

[42] B. Davison, "Content delivery and distribution services," 2003. http://www.web-caching.com/cdns.html.
[43] L. Breslau, P. Cao, L. Fan, G. Philips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. IEEE INFOCOM*, 1999.
[44] W. Fang and L. Peterson, "Inter-AS traffic patterns and their implications," in *Proc. IEEE Global Internet*, 1999.
[45] "Zipf's law." http://linkage.rockefeller.edu/wli/zipf.
[46] B. Krishnamurthy and J. Rexford, *Web Protocols and Practice*. Addison-Wesley, 2001.
[47] V. Padmanabhan and J. Mogul, "Improving HTTP latency," *Computer Networks and ISDN Systems*, 1995.
[48] T. Kroeger, D. Long, and J. Mogul, "Exploring the bounds of Web latency reduction from caching and prefetching," in *Proc. USENIX Symp. on Internet Technologies and Systems*, 1997.
[49] A. Feldmann, "BLT: Bi-Layer Tracing of HTTP and TCP/IP," in *Proc. WWW-9*, 2000.
[50] H. Balakrishnan, V. Padmanabhan, S. Seshan, M. Stemm, and R. Katz, "TCP behavior of a busy Internet server: Analysis and improvements," in *Proc. IEEE INFOCOM*, 1998.
[51] S. Gribble and E. Brewer, "System design issues for Internet middleware services: Deductions from a large client trace," in *Proc. USENIX Symp. on Internet Technologies and Systems*, 1997.
[52] B. Liu, "A different approach to content delivery." http://www.isp-planet.com/news/2001/routescience.html.
[53] V. Paxson, "Bro: A system for detecting network intruders in real-time," in *Computer Networks*, 1999.
[54] RouteViews project. http://www.routeviews.org/.
[55] RIPE's Routing Information Service Raw Data Page. http://data.ris.ripe.net/.
[56] L. Gao, "On inferring autonomous system relationships in the Internet," in *Proc. IEEE Global Internet*, 2000.
[57] A. Feldmann, "Characteristics of TCP connection arrivals," in *Self-Similar Network Traffic And Performance Evaluation* (K. Park and W. Willinger, eds.), J. Wiley & Sons, Inc. 2000.
[58] R. Sommer and A. Feldmann, "Netflow: Information loss or win?," in *Proc. ACM Measurement Workshop*, 2002.

APPENDIX

In this section we present more details on how we estimate publisher demands and web traffic demands using logs from a CDN provider, packet-level measurements at ingress links, and the DNS system.

A. CDN LOG EVALUATION

To compute publisher demands using CDNs, fine-grain access records from all servers of the CDN have to be collected. Usually servers generate a record summarizing each transaction. These are exported on a regular basis for billing purposes and include sufficient information for computing the publisher demand: the `accessed_object`, the `client` IP address, the `start` and `end` times of the transfer, and the number of `transferred_bytes`. (Any additional information can be used to further refine the notion of publisher demands.)

Computing the traffic demands requires information about the CDN customer (i.e., publisher) associated with each record. This aggregation process draws on a map, `object_to_customerid`, such that every `object` can be associated with a unique `customerid`. Furthermore, it uses another map, `clientip_to_clientprefix`, of network addresses such that every source IP address, `client`, can be associated with a network prefix `client_prefix`. The first map can be derived from the customer information of the CDN while the second can be derived with longest prefix match from a joined BGP routing table `joined_bgp_table` from multiple different viewpoints in the Internet or one can use static groups such as up to the /24 level, which (given that most ISP will not allow propagation of prefixes smaller than /19s) does not hinder any later application specific aggregation.

```
For each accessed_object: (client, start, end, transferred_bytes)
    customerid = object_to_customerid(accessed_object);
    clientprefix = longest_prefix_match(client, joined_bgp_table);
    start_bin = ⌊start/bin_length⌋ * bin_length;
    end_bin = ⌊end/bin_length⌋ * bin_length;
    if (start_bin == end_bin)
        volume[clientprefix, customerid, start_bin]
                                        += transferred_bytes;
    else /* Compute volume of traffic for each time_ bin */
        byte_rate = transferred_bytes/(end - start);
        volume[clientprefix, customerid, start_bin]
                            += byte_rate * (start_bin + bin_length - start);
        for (time_bin = start_bin + bin_length; time_bin < end_bin;
                                        time_bin += bin_length)
            volume[clientprefix, customerid, start_bin]
                                        += byte_rate * width;
        volume[clientprefix, customerid, end_bin]
                                        += byte_rate * (end - end_bin);
For each aggregate:
    demand[clientprefix, customerid, end_bin] =
                                    customerid_to_demand[customerid] *
                                    volume[clientprefix, customerid, end_bin]
Output for each aggregate: (clientprefix, customerid, time_bin, demand)
```

Figure 19: Estimating CDN publisher demands from CDN transaction logs.

No content transfer is instantaneous. Rather, they last for some time interval starting at `start`, ending at `end`, and contributing some amount of traffic, `transferred_bytes`. In order to avoid problems in time resolution, e.g., discrepancies between clocks at the record collectors, granularity of the data sources, etc., and since most applications making use of publisher demands are on a larger time scale, we compute the demands on time scales of multiples of minutes rather than seconds. Time is partitioned in bins of duration `bin_length`, according to the considered resolution. If a record spans multiple bins, we subdivide the traffic in proportion to the fraction of time spent in each time period.

To derive the final publisher demands we draw on another map, `customerid_to_demand`. It specifies for each customerid the relationship between the CDN-hosted traffic flows and the self-hosted traffic and is the result of the computation detailed in Section B. The algorithm for computing the publisher demands in summarized in Figure 19.

B. ESTIMATING FLOW RATIOS BETWEEN CDN AND PUBLISHER

In Section 5 we suggest using proxy and/or packet level traces to estimate the relationships between the various flows shown in Figure 4(b). Here we present a three pass approach which automatically ensures that Web pages referring to other Web pages are handled appropriately.

The first two passes serve preparative purposes. In the first pass we separate the set of accessed objects according to users IP addresses. In the second pass (Fig. 20) we determine the set of objects served by the CDN under consideration, `cdn_set`, and some additional information that we specify below. For this purpose we check each object against the appropriate CDN customer base information `determine_customer_id()` and, if appropriate, compute the CDN `customerid` and add it to the `cdn_set`.

In the third pass we compute for each CDN object `cdn_id` within

```
Pass 1:
Sort the accessed objects according to user IP addresses
Pass 2:
For each user IP and object_id: (url, start, end, trans_bytes, referrer, hostname)
    if (determine_customer_id(object_id) evaluates to CDN object) then {
        customerid[object_id] = determine_customer_id(object_id);
        cdn_set ∪= object_id;
    }
    base_candidate_set[url] ∪= object_id;
    embedded_candidate_set[url] ∪= object_id;
Pass 3:
For each object_id from cdn_set
                    with (url, start, end, trans_bytes, referrer, hostname)
    if (done[object_id]) then next;
    done[object_id] = true;
    end_bin_cdn = ⌊end/bin_length⌋ * bin_length;
    cdn_customer_id = customerid[object_id];
    volume[cdn_customer_id, end_bin_cdn] ∪= trans_bytes;
    For each candidate in (base_candidate_set[referrer]
                        or embedded_candidate_set[referrer]) {
        if (∃ customerid[candidate] or done[candidate]) then next;
        done[candidate] = true;
        associated_hosts[cdn_customer_id] ∪= hostname[candidate]
        end_bin_candidate = ⌊end[candidate]/bin_length⌋ * bin_length;
        volume_related[cdn_customer_id, hostname[candidate],
                        end_bin_candidate] ∪= trans_bytes;
    }
Output for each customerid and host from the associated_hosts the ratios:
    (customerid, hostname, time_bin, volume[customerid, time_bin],
    volume_related[host, time_bin]/volume[customerid, time_bin])
```

Figure 20: Computing flow ratios: CDN vs. Publisher from user access logs.

this set the possible base pages `base_candidate_set` and the possible other embedded objects `embedded_candidate_set`. For an object to fall into these sets either its URL or its referrer has to be equal to the referrer value of the CDN object. For this purpose we stored some additional information in the second pass: each object with URL `url` and referrer `referrer` is added to the set of possible home pages for this URL `base_set(url)`. Furthermore, we add the object to the set of possible embedded objects for the current referrer `embedded_set(referrer)`. Once we have retrieved the candidate sets, we can determine the hostnames for each of the objects within the candidate sets and add the bytes in the corresponding object to the appropriate traffic flow. The appropriate traffic flow is either determined by the `cdn_customer_id` for CDN objects or the hostname for non-CDN objects. If the hostname is not used in the users request, we propose to use the server IP address instead. In order to keep the relationship information, we can now establish the link `associated_hosts` between `cdn_customer_id` and the hostname of the objects in the candidate sets. In order to avoid double counting, e.g., if the exact same page is accessed multiple times, one needs to mark every object that has already been accounted for.

Again it is the case that no content transfer is instantaneous, but rather than spreading the contribution of each transfer across multiple time periods of duration `bin_length`, we propose to just add it to the last bin. It is known [58] from aggregating Netflow data that this can lead to artifacts. But if the aggregation periods are long enough, size and impact of these artifacts decrease significantly.

C. MAPPING PUBLISHER DEMANDS TO WEB TRAFFIC DEMANDS

In order to map the publisher demands to Web traffic demands we need to find out which IP addresses are actually in use by the publisher's infrastructure. As an initial step, we derive the set of

```
For each customer_id:
    hostname_set = customerid_to_hostname(customer_id);
    For each host in (hostname_set) {
        For each dns_server in (dns_server_set) {
            ip_set[customer_id] ∪= dns_query(dns_server, host);
            ip_set_dns[customer_id, dns_server]
                                    ∪= dns_query(dns_server, host);
            dns_policy[customer_id] = classify_dns_policy(ip_set)
        }
    }
For each client_prefix:
    closest_dns_server[client_prefix] = closest(client_prefix, dns_server_set);
For each customer_id and client_prefix:
    if (dns_policy[customer_id] == "round robin")
        split traffic evenly among ip_set[customer_id]
    if (dns_policy[customer_id] != "round robin")
        split traffic evenly among
            ip_set_dns[customer_id, closest_dns_server[client_prefix]]
```

Figure 21: Mapping site publishers to Web traffic demands.

hostnames associated with each site publisher (`customer_id`) (via the mapping `customerid_to_hostname`), utilizing the knowledge of the CDN provider. Therefore the problem is reduced to associating each hostname (`host`) with its set of IP addresses (`ip_set`).

To account for the distributed infrastructure of the site we have to issue recursive DNS queries from a set of DNS servers distributed throughout the Internet. We propose identifying a set of candidate DNS servers from traffic measurements, such as Netflow or packet level traces, or by checking Akamai's DNS server logs. Using packet traces has the advantage that its easy to check if the DNS servers support recursive DNS queries. Otherwise one can issue a recursive query to the DNS server and see if it is willing to respond to the query and second if it supports recursive queries. Once we have derived a candidate set of DNS servers, we can either use all of them or a subset. We propose to concentrate on a subset such that each DNS server in the subset will return a different IP address for at least one Web site publisher that utilizes a distributed infrastructure. Since the CDN runs a highly distributed infrastructure we use the main Web server of the CDN, `www.cdn.ex`, for this purpose.

The next step involves identifying what kind of access distribution mechanism (`dns_policy`) is used by the physical Web site. We propose to concentrate on the popular mechanisms and look for indications of their use. If all queried DNS servers return almost the same set of IP addresses then we can assume that DNS round robin ("`round robin`") is used. We use "almost" instead of "exactly" since one cannot query all DNS servers at the same time. This lack of synchrony can cause anomalies. If different DNS servers return different IP addresses in a consistent fashion (at least two times) then we can assume that some form of proximity-aware load balancing is used ("`proximity`"). In the first case we propose to split the load evenly between all IP addresses used to implement the physical infrastructure. Otherwise we propose to split the traffic only between the IP addresses resolved by the closest DNS server queried to the users in question. All other cases are currently resolved via manual inspection.