An Analysis of Internet Content Delivery Systems

Stefan Saroiu, Krishna P. Gummadi, Richard J. Dunn, Steven D. Gribble, and Henry M. Levy
Department of Computer Science & Engineering
University of Washington
{tzoompy,gummadi,rdunn,gribble,levy}@cs.washington.edu

Abstract

In the span of only a few years, the Internet has experienced an astronomical increase in the use of specialized content delivery systems, such as content delivery networks and peer-to-peer file sharing systems. Therefore, an understanding of content delivery on the Internet now requires a detailed understanding of how these systems are used in practice.

This paper examines content delivery from the point of view of four content delivery systems: HTTP web traffic, the Akamai content delivery network, and Kazaa and Gnutella peer-to-peer file sharing traffic. We collected a trace of all incoming and outgoing network traffic at the University of Washington, a large university with over 60,000 students, faculty, and staff. From this trace, we isolated and characterized traffic belonging to each of these four delivery classes. Our results (1) quantify the rapidly increasing importance of new content delivery systems, particularly peer-to-peer networks, (2) characterize the behavior of these systems from the perspectives of clients, objects, and servers, and (3) derive implications for caching in these systems.

1 Introduction

Few things compare with the growth of the Internet over the last decade, except perhaps its growth in the last several years. A key challenge for Internet infrastructure has been delivering increasingly complex data to a voracious and growing user population. The need to scale has led to the development of thousand-node clusters, global-scale content delivery networks, and more recently, self-managing peer-to-peer structures. These content delivery mechanisms are rapidly changing the nature of Internet content delivery and traffic; therefore, an understanding of the modern Internet requires a detailed understanding of these new mechanisms and the data they serve.

This paper examines content delivery by focusing on four content delivery systems: HTTP web traffic, the Akamai content delivery network, and the Kazaa and Gnutella peer-to-peer file sharing systems. To perform the study, we traced all incoming and outgoing Internet traffic at the University of Washington, a large university with over 60,000 students, faculty, and staff. For this paper, we analyze a nine day trace that saw over 500 million transactions and over 20 terabytes of HTTP data. From this data, we provide a detailed characterization and comparison of content delivery systems, and in particular, the latest peer-to-peer workloads. Our results quantify: (1) the extent to which peer-to-peer traffic has overwhelmed web traffic as a leading consumer of Internet bandwidth, (2) the dramatic differences in the characteristics of objects being transferred as a result, (3) the impact of the two-way nature of peer-to-peer communication, and (4) the ways in which peer-to-peer systems are *not* scaling, despite their explicitly scalable design. For example, our measurements show that an average peer of the Kazaa peer-to-peer network consumes 90 times more bandwidth than an average web client in our environment. Overall, we present important implications for large organizations, service providers, network infrastructure, and general content delivery.

The paper is organized as follows. Section 2 presents an overview of the content delivery systems examined in this paper, as well as related work. Section 3 describes the measurement methodology we used to collect and process our data. In Section 4 we give a high-level overview of the workload we have traced at the University of Washington. Section 5 provides a detailed analysis of our trace from the perspective of objects, clients, and servers, focusing in particular on a comparison of peer-to-peer and web traffic. Section 6 evaluates the potential for caching in content delivery networks and peer-to-peer networks, and Section 7 concludes and summarizes our results.

2 Overview of Content Delivery Systems

Three dominant content delivery systems exist today: the client/server oriented world-wide web, content delivery networks, and peer-to-peer file sharing systems. At a high level, these systems serve the same role of distributing content to users. However, the architectures of these systems differ significantly, and the differences affect their performance, their workloads, and the role caching can play. In this section, we present the architectures of these systems and describe previous studies of their behavior.

2.1 The World-Wide Web (WWW)

The basic architecture of the web is simple: using the HTTP [16] protocol, web clients running on users' machines request objects from web servers. Previous studies have examined many aspects of the web, including web workloads [2, 8, 15, 29], characterizing web objects [3, 11], and even modeling the hyperlink structure of the web [6, 21]. These studies suggest that most web objects are small (5-10KB), but the distribution of object sizes is heavy-tailed and very large objects exist. Web objects are accessed with a Zipf popularity distribution, as are web servers. The number of web objects is enormous (in the billions) and rapidly growing; most web objects are static, but an increasing number are generated dynamically.

The HTTP protocol includes provisions for consistency management. HTTP headers include caching pragmas that affect whether or not an object may be cached, and if so, for how long. Web caching helps to alleviate load on servers and backbone links, and can also serve to decrease object access latencies. Much research has focused on Web proxy caching [4, 5, 7, 11, 12] and, more recently, on coordinating state among multiple, cooperating proxy caches [13, 30, 33]; some of these proposals aim to create global caching structures [27, 34]. The results of these studies generally indicate that cache hit rates of 40-50% are achievable, but that hit rate increases only logarithmically with client population [36] and is constrained by the increasing amount of dynamically generated and hence uncacheable content.

2.2 Content Delivery Networks (CDNs)

Content delivery networks are dedicated collections of servers located strategically across the wide-area Internet. Content providers, such as web sites or streaming video sources, contract with commercial CDNs to host and distribute content. CDNs are compelling to content providers because the responsibility for hosting content is offloaded to the CDN infrastructure. Once in a CDN, content is replicated across the wide area, and hence is highly available. Since most CDNs have servers in ISP points of presence, clients can access topologically nearby replicas with low latency. The largest CDNs have thousands of servers dispersed throughout the Internet and are capable of sustaining large workloads and traffic hot-spots.

CDNs are tightly integrated into the existing web architecture, relying either on DNS interposition [19, 32] or on URL rewriting at origin servers to redirect HTTP requests to the nearest CDN replica. As with the web, the unit of transfer in a CDN is an object, and objects are named by URLs. Unlike the web, content providers need not manage web servers, since clients' requests are redirected to replicas hosted by the CDN. In practice, CDNs typically host static content such as images, advertisements, or media clips; content providers manage their own dynamic content, although dynamically generated web pages might contain embedded objects served by the CDN.

Previous research has investigated the use and effectiveness of content delivery networks [14], although the proprietary and closed nature of these systems tends to impede investigation. Two recent studies [22, 23] confirm that CDNs reduce average download response times, but that DNS redirection techniques add noticeable overhead because of DNS latencies. In another study [18], the authors argue that the true benefit of CDNs is that they help clients avoid the worst-case of badly performing replicas, rather than routing clients to a truly optimal replica. To the best of our knowledge, no study has yet compared the workloads of CDNs with other content delivery architectures.

2.3 Peer-to-Peer Systems (P2P)

Peer-to-peer file sharing systems have surged in popularity in recent years. In a P2P system, peers collaborate to form a distributed system for the purpose of exchanging content. Peers that connect to the system typically behave as servers as well as clients: a file that one peer downloads is often made available for upload to other peers. Participation is purely voluntary, and a recent study [31] has shown that most content-serving hosts are run by end-users, suffer from low availability, and have relatively low capacity network connections (modem, cable modems, or DSL routers).

Users interact with a P2P system in two ways: they attempt to locate objects of interest by issuing search queries, and once relevant objects have been located, users issue download requests for the content. Unlike the web and CDN systems, the primary mode of usage for P2P systems is a non-interactive, batch-style download of content.

P2P systems differ in how they provide search capabilities to clients [37]. Some systems, such as Napster [28], have large, logically centralized indexes maintained by a single company; peers automatically upload lists of available files to the central index, and queries are answered using this index. Other systems, such as Gnutella [10] and Freenet [9], broadcast search requests over an overlay network connecting the peers. More recent P2P systems, including Kazaa [20], use a hybrid architecture in which some peers are elected as "supernodes" in order to index content available at peers in a nearby neighborhood.

P2P systems also differ in how downloads proceed, once an object of interest has been located. Most systems transfer content over a direct connection between the object provider and the peer that issued the download request. A latency-improving optimization in some systems is to download multiple object fragments in parallel from multiple replicas. A recent study [24] has found the peer-to-peer traffic of a small ISP to be highly repetitive, showing great potential for caching.

3 Methodology

We use *passive network monitoring* to collect traces of traffic flowing between the University of Washington (UW) and the rest of the Internet. UW connects to its ISPs via two border routers; one router handles outbound traffic and the other inbound traffic. These two routers are fully connected to four switches on each of the four campus backbones. Each switch has a monitoring port that is used to send copies of the incoming and outgoing packets to our monitoring host.

Our tracing infrastructure is based on software developed by Wolman and Voelker for previous studies [35, 36]. We added several new components to identify, capture, and analyze Kazaa and Gnutella peer-to-peer traffic and Akamai CDN traffic. Overall, the tracing and analysis software is approximately 26,000 lines of code. Our monitoring host is a dual-processor Dell Precision Workstation 530 with 2.0 GHz Pentium III Xeon CPUs, a Gigabit Ethernet SysKonnect SK-9843 network card, and running FreeBSD 4.5.

Our software installs a kernel packet filter [26] to deliver TCP packets to a user-level process. This process reconstructs TCP flows, identifies HTTP requests within the flows (properly handling persistent HTTP connections), and extracts HTTP headers and other metadata from the flows. Because Kazaa and Gnutella use HTTP to exchange files, this infrastructure is able to capture P2P downloads as well as WWW and Akamai traffic. We anonymize sensitive information such as IP addresses and URLs, and log all extracted data to disk in a compressed binary representation.

3.1 Distinguishing Traffic Types

Our trace captures two types of traffic: *HTTP traffic*, which can be further broken down into WWW, Akamai, Kazaa, and Gnutella transfers, and *non-HTTP TCP traffic*, including Kazaa and Gnutella search traffic. If an HTTP request is directed to port 80, 8080, or 443 (SSL), we classify both the request and the associated response as WWW traffic. Similarly, we use ports 6346 and 6347 to identify Gnutella HTTP traffic, and port 1214 to identify Kazaa HTTP traffic. A small part of our captured HTTP traffic remains unidentifiable; we believe that most of this traffic can be attributed to less popular peer-to-peer systems (e.g., Napster [28]) and by compromised hosts turned into IRC or web servers on ports other than 80, 8080, or 444. For non-HTTP traffic, we use the same Gnutella and Kazaa ports to identify P2P search traffic.

Some WWW traffic is served by the Akamai content delivery network [1]. Akamai has deployed over 13,000 servers in more than 1,000 networks around the world [25]. We identify Akamai traffic as any HTTP traffic served by any Akamai server. To obtain a list of Akamai servers, we collected a list of 25,318 unique authoritative name servers, and sent a recursive DNS query to each server for a name in an Akamai-managed domain (e.g., `a388.g.akamaitech.net`). Because Akamai redirects DNS queries to nearby Akamai servers, we were able to collect a list of 3,966 unique Akamai servers in 928 different networks.

For the remainder of this paper, we will use the following definitions when classifying traffic:

- **Akamai:** HTTP traffic on port 80, 8080, or 443 that is served by an Akamai server.
- **WWW:** HTTP traffic on port 80, 8080, or 443 that is not served by an Akamai server; thus, for all of the analysis within this paper, "WWW traffic" does not include Akamai traffic.
- **Gnutella:** HTTP traffic sent to ports 6346 or 6347 (this includes file transfers, but excludes search and control traffic).
- **Kazaa:** HTTP traffic sent to port 1214 (this includes file transfers, but excludes search and control traffic).
- **P2P:** the union of Gnutella and Kazaa.
- **non-HTTP TCP traffic:** any other TCP traffic, including protocols such as NNTP and SMTP, HTTP traffic to ports other than those listed above, traffic from other P2P systems, and control or search traffic on Gnutella and Kazaa.

3.2 The Traceability of P2P Traffic

Gnutella is an overlay network over which search requests are flooded. Peers issuing search requests receive a list of other peers that have matching content. From this list, the peer that issued the request initiates a direct connection with one of the matching peers to download content. Because the Gnutella overlay is not structured to be efficient with respect to the physical network topology, most downloads initiated by UW peers connect to external hosts, and are therefore captured in our traces.

Although the details of Kazaa's architecture are proprietary, some elements are known. The Kazaa network is a two-level overlay: some well-connected peers serving as "supernodes" build indexes of the content stored on nearby "regular" peers. To find content, regular peers issue search requests to their supernodes. Supernodes appear to communicate amongst themselves to satisfy queries, returning locations of matching objects to the requesting peer. Kazaa appears to direct peers to nearby objects, although the details of how this is done, or how successful the system is at doing it, are not known.

To download an object, a peer initiates one or more connections to other peers that have replicas of the object. The downloading peer may transfer the entire object in one connection from a single peer, or it may choose to download multiple fragments in parallel from multiple peers.

	WWW		Akamai		Kazaa		Gnutella	
	inbound	outbound	inbound	outbound	inbound	outbound	inbound	outbound
HTTP transactions	329,072,253	73,001,891	33,486,508	N/A	11,140,861	19,190,902	1,576,048	1,321,999
unique objects	72,818,997	3,412,647	1,558,852	N/A	111,437	166,442	5,274	2,092
clients	39,285	1,231,308	34,801	N/A	4,644	611,005	2,151	25,336
servers	403,087	9,821	350	N/A	281,026	3,888	20,582	412
bytes transferred	1.51 TB	3.02 TB	64.79 GB	N/A	1.78 TB	13.57 TB	28.76 GB	60.38 GB
median object size	1,976 B	4,646 B	2,001 B	N/A	3.75 MB	3.67 MB	4.26 MB	4.08 MB
mean object size	24,687 B	82,385 B	12,936 B	N/A	27.78 MB	19.07 MB	19.16 MB	9.78 MB

Table 1. HTTP trace summary statistics: trace statistics, broken down by content delivery system; *inbound* refers to transfers from Internet servers to UW clients, and *outbound* refers to transfers from UW servers to Internet clients. Our trace was collected over a nine day period, from Tuesday May 28th through Thursday June 6th, 2002.

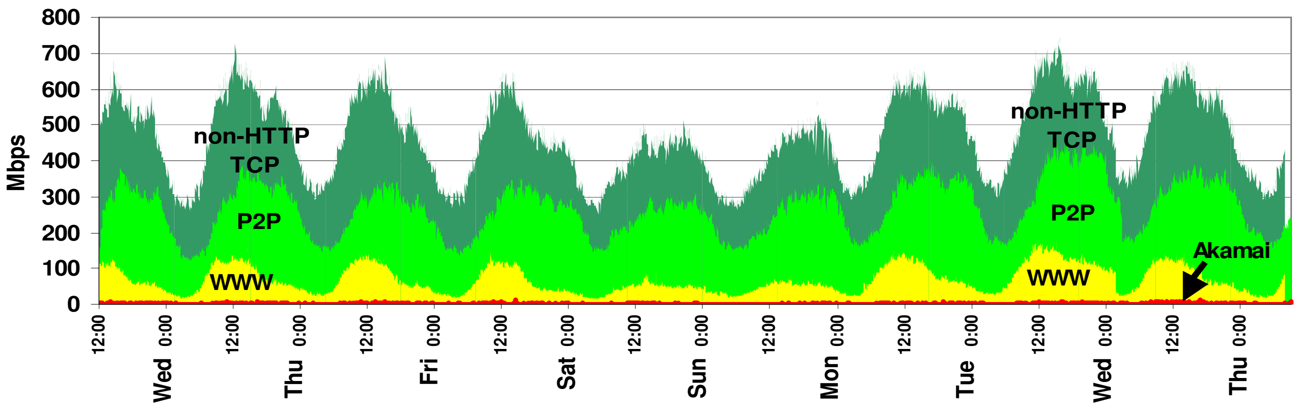


Figure 1. TCP bandwidth: total TCP bandwidth consumed by HTTP transfers for different content delivery systems. Each band is cumulative; this means that at noon on the first Wednesday, Akamai consumed approximately 10 Mbps, WWW consumed approximately 100 Mbps, P2P consumed approximately 200 Mbps, and non-HTTP TCP consumed approximately 300 Mbps, for a total of 610 Mbps.

The ability for a Kazaa peer to download an object in fragments complicates our trace. Download requests from external peers seen in our trace are often for fragments rather than entire objects.

4 High-Level Data Characteristics

This section presents a high-level characterization of our trace data. Table 1 shows summary statistics of object transfers. This table separates statistics from the four content delivery systems, and further separates inbound data (data requested by UW clients from outside servers) from outbound data (data requested by external clients from UW servers).

Despite its large client population, the University is a net *provider* rather than consumer of HTTP data, exporting 16.65 TB but importing only 3.44 TB. The peer-to-peer systems, and Kazaa in particular, account for a large percentage of the bytes exported and the total bytes transferred, despite their much smaller internal and external client populations. Much of this is attributable to a large difference in average object sizes between WWW and P2P systems.

The number of clients and servers in Table 1 shows the extent of participation in these systems. For the web, 39,285 UW clients accessed 403,437 Internet web servers, while for Kazaa, 4,644 UW clients accessed 281,026 external Internet servers. For Akamai, 34,801 UW clients download Akamai-hosted content provided by 350 different Akamai servers. In the reverse direction, 1,231,308 Internet clients accessed UW web content, while 611,005 clients accessed UW-hosted Kazaa content.

Figure 1 shows the total TCP bandwidth consumed in both directions over the trace period. The shaded areas show HTTP traffic, broken down by content delivery system; Kazaa and Gnutella traffic are grouped together under the label "P2P." All systems show a typical diurnal cycle. The smallest bandwidth consumer is Akamai, which currently constitutes only 0.2% of observed TCP traffic. Gnutella consumes 6.04%, and WWW traffic is the next largest, consuming 14.3% of TCP traffic. Kazaa is currently the largest contributor, consuming 36.9% of TCP bytes. These four content delivery systems account for 57% of total TCP traffic, leaving 43% for other TCP-based network protocols (streaming media, news, mail, and so on). TCP traffic represents over 97% of all network traffic at UW. This closely matches published data on Internet 2 usage [17].

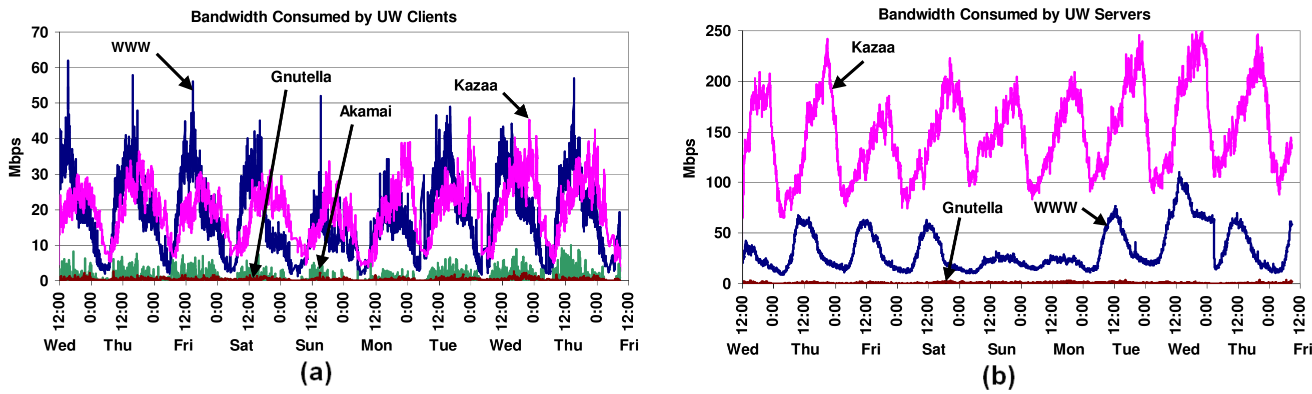


Figure 2. UW client and server TCP bandwidth: bandwidth over time (a) accountable to web and P2P downloads from UW clients, and (b) accountable to web and P2P uploads from UW servers.

Figures 2a and 2b show inbound and outbound data bandwidths, respectively. From Figure 2a we see that while both WWW and Kazaa have diurnal cycles, the cycles are offset in time, with WWW peaking in the middle of the day and Kazaa peaking late at night. For UW-initiated requests, WWW and Kazaa peak bandwidths have the same order of magnitude; however, for requests from external clients to UW servers, the peak Kazaa bandwidth dominates WWW by a factor of three. Note that the Y-axis scales of the graphs are different; WWW peak bandwidth is approximately the same in both directions, while external Kazaa clients consume 7.6 times more bandwidth than UW Kazaa clients.

Figure 3a and 3b show the top 10 content types requested by UW clients, ordered by bytes downloaded and number of downloads. While GIF and JPEG images account for 42% of requests, they account for only 16.3% of the bytes transferred. On the other hand, AVI and MPG videos, which account for 29.3% of the bytes transferred, constitute only 0.41% of requests. HTML is significant, accounting for 14.6% of bytes and 17.8% of requests. The 9.9% of bytes labelled "HASHED" in Figure 3a are Kazaa transfers that cannot be identified; of the non-hashed Kazaa traffic that can be identified, AVI and MPG account for 79% of the bytes, while 13.6% of the bytes are MP3.

It is interesting to compare these figures with corresponding measurements from our 1999 study of the same population [35]. Looking at bytes transferred as a percent of total HTTP traffic, HTML traffic has decreased 43% and GIF/JPG has decreased 59%. At the same time, AVI/MPG (and Quicktime) traffic has increased by nearly 400%, while MP3 traffic has increased by nearly 300%. (These percentages numbers include an estimate of the appropriate portion of the hashed bytes contributing to all content types).

In summary, this high-level characterization reveals substantial changes in content delivery systems usage in the Internet, as seen from the vantage point of UW. First, the balance of HTTP traffic has changed dramatically over the last several years, with P2P traffic overtaking WWW traffic as the largest contributor to HTTP bytes transferred. Second, although UW is a large publisher of web documents,

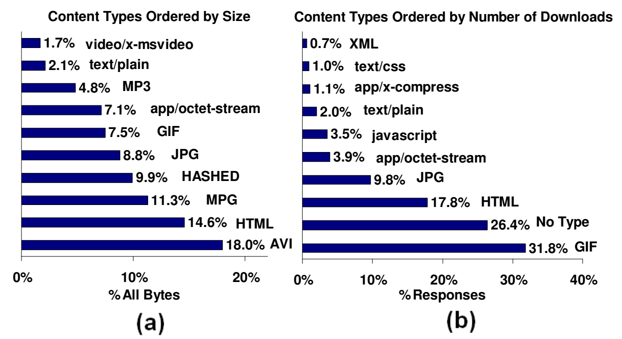


Figure 3. Content types downloaded by UW clients: a histogram of the top 10 content types downloaded by UW clients, across all four systems, ordered by (a) size and (b) number of downloads.

P2P traffic makes the University an even larger exporter of data. Finally, the mixture of object types downloaded by UW clients has changed, with video and audio accounting for a substantially larger fraction of traffic than three years ago, despite the small number of requests involving those data types.

5 Detailed Content Delivery Characteristics

The changes in Internet workload that we have observed raise several questions, including: (1) what are the properties of the new objects being delivered, (2) how are clients using the new content-delivery mechanisms, and (3) how do servers for new delivery services differ from those for the web? We attempt to answer these questions in the subsections below.

5.1 Objects

Data in Section 4 suggests that there is a substantial difference in typical object size between P2P and WWW traffic. Figure 4 illustrates this in dramatic detail. Not surprisingly, Akamai and WWW object sizes track each other fairly closely. The median WWW object is approximately 2KB, which matches previous measurement studies [15]. The Kazaa and Gnutella curves are strikingly different from the WWW; the median object size for these P2P systems is

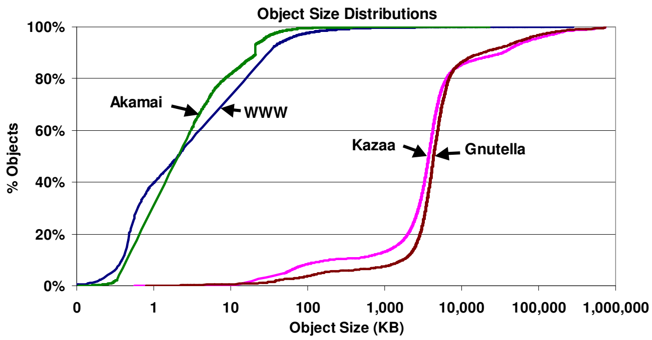


Figure 4. Object size distributions: cumulative distributions (CDFs) of object sizes.

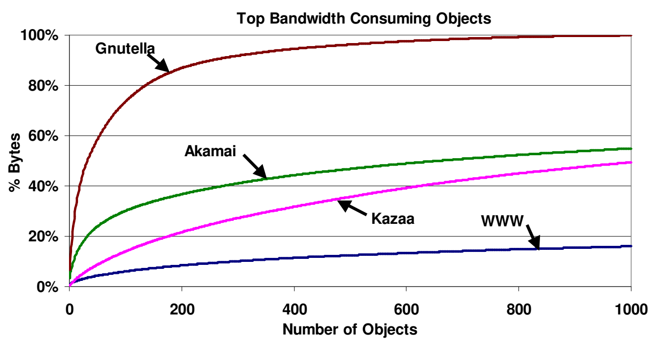


Figure 5. Top bandwidth consuming objects: a CDF of bytes fetched by UW clients for top 1,000 bandwidth-consuming objects.

approximately 4MB – a *thousand-fold* increase over the average web document size! Worse, we see that 5% of Kazaa objects are over 100MB. This difference has the potential for enormous impact on Internet performance as these systems grow.

Figure 5 shows a cumulative distribution of bytes fetched by UW clients for the 1,000 highest bandwidth-consuming objects in each of the four CDNs. The Akamai curve rises steeply, with the top 34 objects accounting for 20% of the Akamai bytes transferred; Akamai traffic is clearly skewed to its most popular documents. For Kazaa, we see that a relatively small number of objects account for a large portion of the transferred bytes as well. The top 1,000 Kazaa objects (out of 111K objects accessed) are responsible for 50% of the bytes transferred. For the web, however, the curve is much flatter: the top 1,000 objects only account for 16% of bytes transferred.

To understand this better, we examined the 10 highest bandwidth-consuming objects for WWW, Akamai and Kazaa, which are responsible for 1.9%, 25% and 4.9% of the traffic for each system, respectively. The details are shown in Table 2. For WWW, we see that the top 10 objects are a mix of extremely popular small objects (e.g., objects 1, 2 and 4), and relatively unpopular large objects (e.g., object 3). The worst offender, object 1, is a small object accessed many times. For Akamai, although 8 out of the top

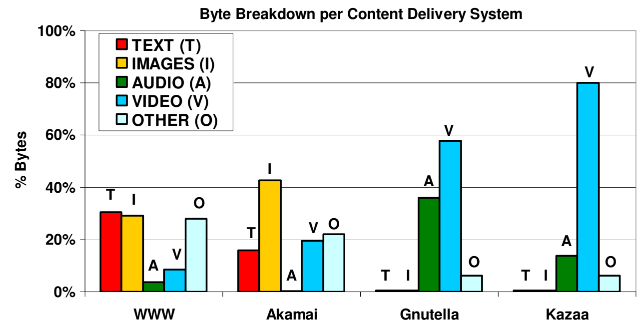


Figure 6. Downloaded bytes by object type: the number of bytes downloaded from each system, broken into content type.

10 objects are large and unpopular, 2 out of the top 3 worst offenders are small and popular. Kazaa's inbound traffic, on the other hand, is completely consistent; all of its worst offenders are extremely large objects (on the order of 700MB) that are accessed only ten to twenty times.

Comparing Kazaa inbound and outbound traffic in Table 2 shows several differences. The objects that contribute most to bandwidth consumption in either direction are similarly sized, but UW tends to export these large objects more than it imports them. A small number of UW clients access large objects from a small number of external servers, but nearly thirty times as many external clients access similarly-sized objects from a handful of UW servers, leading to approximately ten times as much bandwidth consumption. This suggests that a *reverse* cache that absorbs outbound traffic might benefit the University even more than a forward cache that absorbs inbound traffic.

Figure 3 in the previous section showed a breakdown of all HTTP traffic by content type for UW-client-initiated traffic. Figure 6 shows a similar breakdown, by bytes, but for each individual CDN. Not surprisingly, the highest component of WWW traffic is text, followed closely by images, while Akamai is dominated by images (42% of bytes are GIF and JPEG). In contrast, Kazaa is completely dominated by video (80%), followed by 14% audio; Gnutella is more evenly split with 58% video and 36% audio.

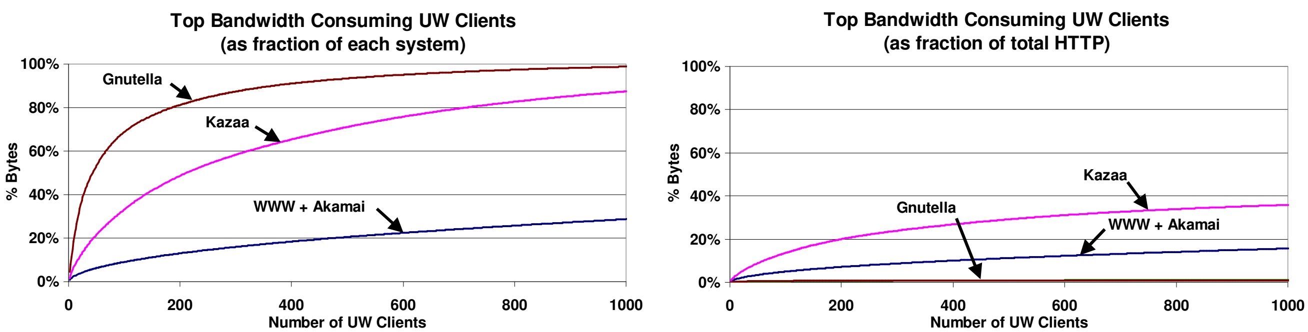
5.2 Clients

The previous subsection considered the characteristics of *what* is transferred (the object view); here we consider *who* is responsible (the client view). Because WWW and Akamai are indistinguishable from a UW client's perspective, this section presents these two workloads combined.

Figure 7a shows a cumulative distribution of bytes downloaded by the top 1000 bandwidth-consuming UW clients for each CDN. It's not surprising that the WWW+Akamai curve is lower; the graph shows only a small fraction of the 39K WWW+Akamai clients, but nearly a quarter of the 4644 Kazaa clients. Nevertheless, in both cases, a small number of clients account for a large portion of the traffic. In the case of the WWW, the top 200 clients (0.5% of the

	WWW (inbound)			Akamai			Kazaa (inbound)				Kazaa (outbound)			
	object size (MB)	GB consumed	# requests	object size (MB)	GB consumed	# requests	object size (MB)	GB consumed	# clients	# servers	object size (MB)	GB consumed	# clients	# servers
1	0.009	12.29	1,412,104	22.37	4.72	218	694.39	8.14	20	164	696.92	119.01	397	1
2	0.002	6.88	3,007,720	0.07	2.37	45,399	702.17	6.44	14	91	699.28	110.56	1000	4
3	333	6.83	21	0.11	1.64	68,202	690.34	6.13	22	83	699.09	78.76	390	10
4	0.005	6.82	1,412,105	9.16	1.59	2,222	775.66	5.67	16	105	700.86	73.30	558	2
5	2.23	3.17	1,457	13.78	1.31	107	698.13	4.70	14	74	634.25	64.99	540	1
6	0.02	2.69	126,625	82.03	1.14	23	712.97	4.69	17	120	690.34	64.97	533	10
7	0.02	2.69	122,453	21.05	1.01	50	715.61	4.49	13	71	690.34	54.90	447	16
8	0.03	1.92	56,842	16.75	1.00	324	579.13	4.30	14	158	699.75	49.47	171	2
9	0.01	1.91	143,780	15.84	0.95	68	617.99	4.12	12	94	696.42	43.35	384	14
10	0.04	1.86	47,676	15.12	0.80	57	167.18	3.83	39	247	662.69	42.28	151	2

Table 2. Top 10 bandwidth consuming objects: the size, bytes consumed, and number of requests (including the partial and unsuccessful ones) for the top 10 bandwidth consuming objects in each system. For Kazaa, instead of requests, we show the number of clients and servers that participated in (possibly partial) transfers of the object.



(a) (b)

Figure 7. Top UW bandwidth consuming clients: a CDF of bytes downloaded by the top 1000 bandwidth-consuming UW clients (a) as a fraction of each system, and (b) as a fraction of the total HTTP traffic.

population) account for 13% of WWW traffic; for Kazaa, the top 200 clients (4% of the population) account for 50% of Kazaa traffic. The next 200 Kazaa clients account for another 16% of its traffic. Clearly, a very small number of Kazaa clients have a huge overall bandwidth impact.

To see the impact more globally, the curves in Figure 7b show the *fraction* of the total HTTP bytes downloaded by the most bandwidth-consuming clients for each CDN (the curves are cumulative). This allows us to quantify the impact of a particular CDN's clients on total HTTP traffic. Gnutella clients have almost no impact as consumers of HTTP bandwidth. In contrast, the Kazaa users are the worst offenders: the top 200 Kazaa clients are responsible for 20% of the total HTTP bytes downloaded. In comparison, the top 200 WWW+Akamai clients are responsible for only 7% of total HTTP bytes. Further out, the top 400 Kazaa and WWW clients are responsible for 27% and 10% of total HTTP traffic, respectively.

Given the bandwidth consumed by the web and peer-to-peer delivery systems, it is interesting to examine the request rates that are creating that bandwidth. Figures 8a and 8b show the inbound and outbound request rates for WWW+Akamai and Kazaa, respectively; notice the nearly two order-of-magnitude difference in the Y axis scales. For Kazaa, the outbound request rate peaks at 40 requests per second, dominating the inbound request rate of 23 requests per second. In contrast, the WWW+Akamai inbound request rate peaks at 1100 requests per second, dominating the WWW outbound[1] request rate of just under 200 requests per second. At a high level, then, Kazaa has a request rate about two orders of magnitude lower than the web, but median object size about three orders of magnitude higher than the web. The result is that overall, Kazaa consumes more bandwidth. Similarly, WWW's outbound bandwidth exceeds its inbound bandwidth, despite the opposite trend in request rate; this results from the difference in transfer size in the two directions. While inbound web documents are largely HTML or images, outbound is dominated by application/octet-streams (possibly UW-supplied software, binary data, and video streams from its TV station or web-broadcast technical talks).

A perhaps surprising (although now understandable) result of the disparity in WWW+Akamai and Kazaa object sizes and request rates is shown in Figure 9, which graphs the number of *concurrent* HTTP transactions active at a time for the two systems. Despite the order-of-magnitude

[1] No Akamai data is present in the outbound direction.

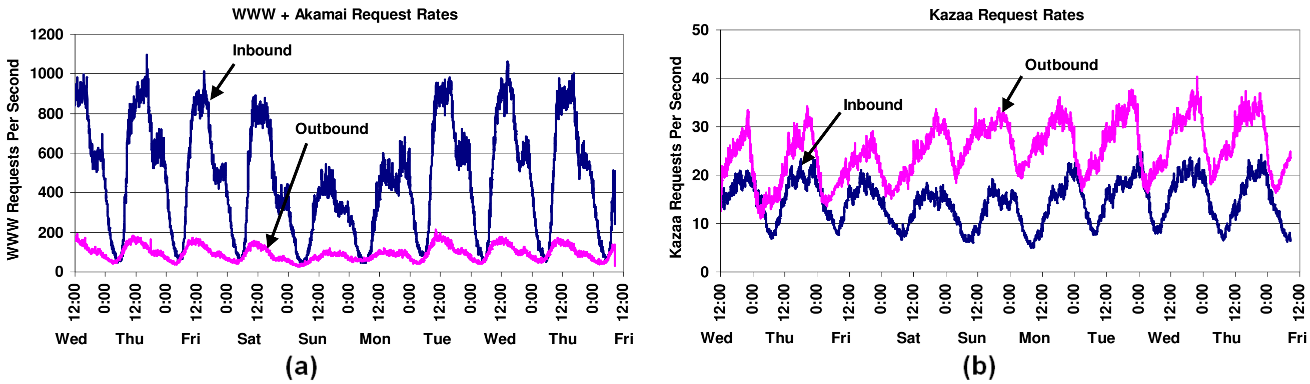


Figure 8. Request rates over time: inbound and outbound HTTP transaction rates for (a) the WWW + Akamai, and (b) Kazaa.

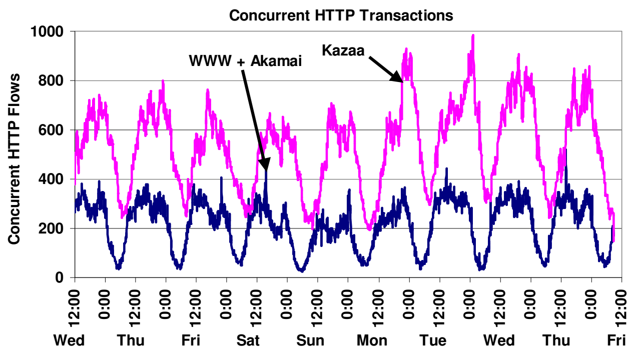


Figure 9. Concurrent HTTP transactions: concurrent HTTP transactions for UW Clients.

request-rate advantage of WWW over Kazaa, the number of simultaneous open Kazaa connections is about twice the number of simultaneous open WWW+Akamai connections. While Kazaa generates only 23 requests per second, it is responsible for up to almost 1000 open requests at a time due to its long transfers. Compared to the web requests, whose median duration is 120 ms, the median duration for Kazaa requests is 130 seconds – a 1000-fold increase that tracks the object size. This fact has important implications for the network infrastructure that must maintain those connections.

5.3 Servers

This section examines servers: the suppliers of objects and bytes. Figure 10a shows the CDF of bytes transferred by UW-internal servers to external clients. Gnutella has the smallest number of content-serving hosts, and all of the bytes are served by the first 10 of those servers. The WWW curve is quite steep; in this case, the campus has several major servers that provide documents to the Web, and 80% of the WWW traffic is supplied by 20 of the 9821 internal servers we identified. The Kazaa curve rises more slowly, with 80% of the Kazaa bytes being served by the top 334 of the 3888 Kazaa peers we found serving data. One would expect the Kazaa curve to be flatter; an explicit goal of peer-to-peer structures like Kazaa is to spread the load uniformly across the peers. We'll look at this issue in more detail with external Kazaa servers.

Figure 10b shows the fraction of total HTTP bytes transferred (cumulative) from the top UW servers for the CDNs. The global impact of a small number of internal Kazaa peers is clearly seen on the graph. Again, a small number of WWW servers do most of the work for WWW, but this is a small part of the total HTTP outgoing traffic; 20 WWW servers provide 20% of all HTTP bytes output, and the curve rises very slowly from there. However, from the Kazaa curve we see that 170 Kazaa peers are responsible for over 50% of all HTTP bytes transferred; the top 400 Kazaa peers are creating 70% of all outgoing HTTP traffic.

In the opposite direction (UW-external servers), Figures 11a and 11b show the cumulative and HTTP-fractional distributions for incoming bytes transferred, respectively, from the top 1,000 external servers to UW clients. The cumulative distribution (Figure 11a) shows the WWW and Kazaa curves rising very slowly. The WWW curve first rises steeply (for the most popular servers), then levels off, with 938 (out of 400,000) external servers supplying 50% of the total WWW bytes to UW clients; this closely matches our previous findings [35]. The Kazaa curve shows that 600 external Kazaa peers (out of 281,026) supply 26% of the Kazaa bytes to internal peers; this result, however, is somewhat unexpected. Web clients request data from *specific* servers by specifying a URL. A small number of web servers are highly popular, and the popularity curve has a large tail (Zipf) of very unpopular servers. Peer-to-peer systems, though, are different by design. Clients request documents by name, not by server. Those documents may (and hopefully, will) exist on many peers. The goal of the peer-to-peer overlay structure is to broadly distribute work for both scalability and availability. In Kazaa, large files are downloaded by transferring different fragments of the file from different peers, to spread the load among multiple peers. Overall, one would expect the server load for Kazaa to be *much* more widely distributed among peers than for WWW. From Figure 11a, this does not appear to be the case.

As a fraction of total HTTP bytes received by UW

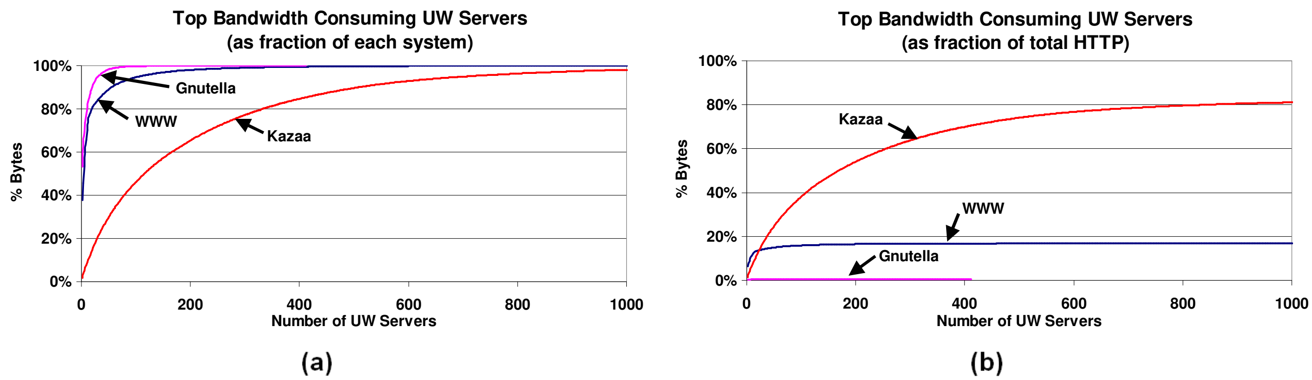


Figure 10. Top UW-internal bandwidth producing servers: a CDF of bytes produced by the top 1000 bandwidth-producing UW-internal servers (a) as a fraction of each system, and (b) as a fraction of the total HTTP traffic.

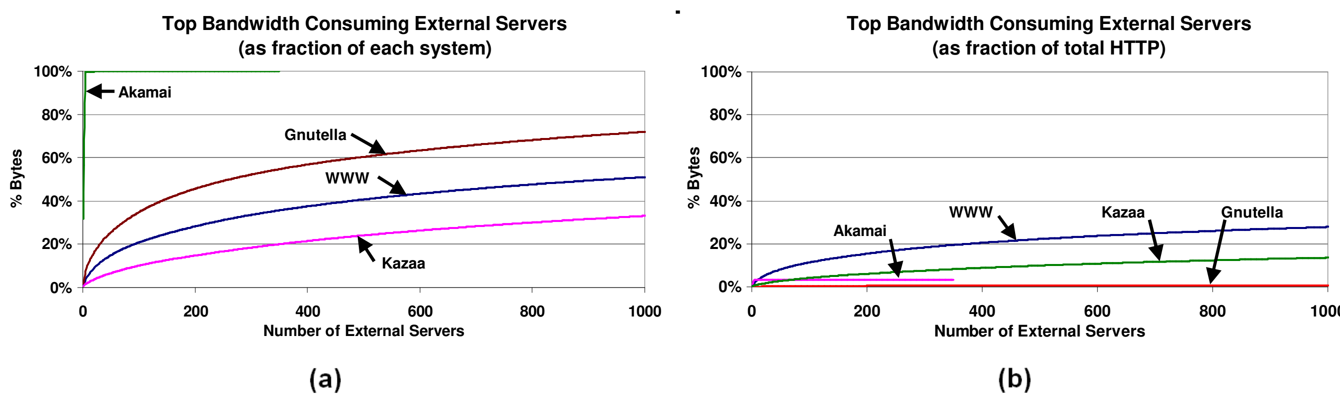


Figure 11. Top UW-external bandwidth producing servers: a CDF of bytes produced by the top 1000 bandwidth-producing UW-external servers (a) as a fraction of each system, and (b) as a fraction of the total HTTP traffic.

clients, Figure 11b shows that the top 500 external Kazaa peers supply 10% of the bytes to UW, while the top 500 WWW servers supply 22% of the bytes. Gnutella and Akamai serve an insignificant percentage of the bytes delivered.

Participation in a P2P system is voluntary, and as a result, servers on P2P system are often less well-provisioned than in the web or a CDN [31]. In Figure 12, we show the response codes returned by external servers in each content delivery system. Figure 12a shows that for Akamai and the WWW, approximately 70% of requests result in a successful transaction. However, for P2P systems, less than 20% of requests result in a successful transaction. Most P2P requests are met with a "service unavailable" response, suggesting that P2P servers are often saturated.

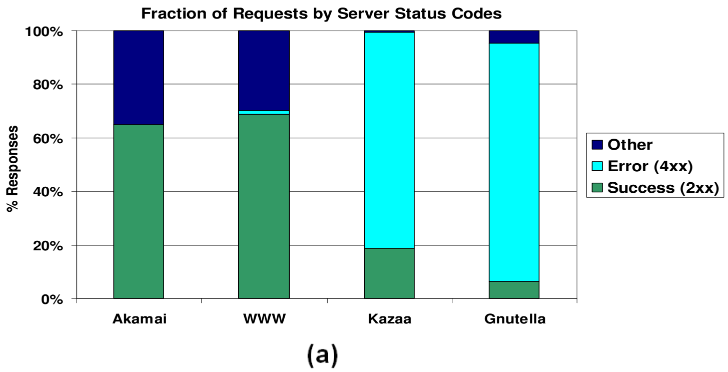
Figure 12b shows that nearly *all* HTTP bytes transferred in WWW, Akamai and P2P systems are for useful content. Even though most P2P requests are rejected, the overhead of rejected requests is small compared to the amount of useful data transferred while downloading content.

5.4 Scalability of Peer-to-Peer Systems

The data presented here raises serious questions about whether P2P systems like Kazaa can scale in environments such as the University. Unlike the web, where most clients and servers are separate entities, every peer in a P2P system consumes bandwidth in both directions. Each new P2P client added becomes an immediate server for the entire P2P structure. In addition, we see that the average Kazaa object is huge, and a small number of peers can consume an enormous amount of total network bandwidth in both directions. Over the period of our trace, an average web client consumed 41.9 MB of bandwidth; in contrast, an average Kazaa peer consumed 3.6 GB of bandwidth. Therefore, the bandwidth cost of each Kazaa peer is approximately 90 times that of a web client! This implies that for our environment, adding another 450 Kazaa peers would be equivalent to *doubling* the entire campus web client population. It seems questionable whether any organization providing bandwidth to a large client population can, in the long run, support a service with these characteristics.

5.5 Summary

This section examined our HTTP workload with respect to objects, clients, and servers. From the data presented, we find several important observations:



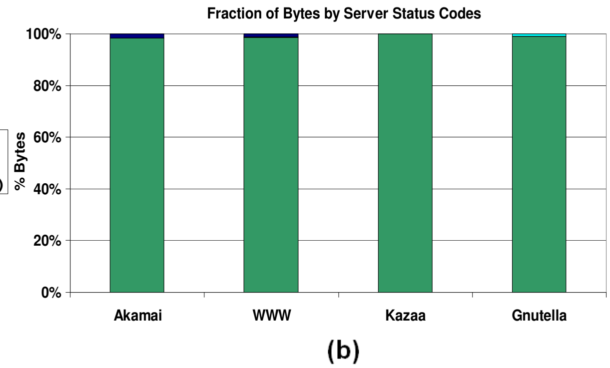


Figure 12. Server status codes: response codes returned by external servers; (a) shows the fraction of requests broken down by response code, (b) shows the fraction of bytes broken down by response code.

1. Peer-to-peer traffic, which now accounts for over three quarters of HTTP traffic in our environment, is characterized by objects whose median sizes are three orders of magnitude larger than web objects. It is this object size, rather than widespread usage, that accounts for the large fraction of bytes transferred due to P2P traffic.

2. A small number of P2P users are consuming a disproportionately high fraction of bandwidth. The 200 top Kazaa clients are responsible for 50% of Kazaa bytes downloaded, and nearly 27% of *all* HTTP bytes received by UW.

3. While the P2P request rate is quite low, the transfers last long (three orders of magnitude longer than WWW transfers), resulting in many simultaneous P2P connections. Despite a two order of magnitude difference between P2P and WWW request rates, the number of simultaneous open P2P connections is twice the number of simultaneous open WWW connections.

4. While the design of P2P overlay structures focuses on spreading the workload for scalability, our measurements show that a small number of servers are taking the majority of the burden. In our measurements, only 600 of the 281,026 UW-external Kazaa peers we saw provided 26% of the bytes received by UW clients.

These results have implications for the web as a whole, for the scalability of peer-to-peer systems, and for the potentials for the use of caching in our environment. We focus on this latter issue in the next section.

6 The Potential Role of Caching in CDNs and P2P Systems

Caching in the web is well understood: caches have been shown to absorb bandwidth and reduce access latency [4, 5, 7, 11, 12]. In this section of the paper, we explore caching in the context of the Akamai CDN and Kazaa P2P systems. For Akamai, we ask whether or not there is a performance advantage of an Akamai CDN relative to a local proxy cache, and for Kazaa, we present an initial study of the effectiveness of caching in that environment.

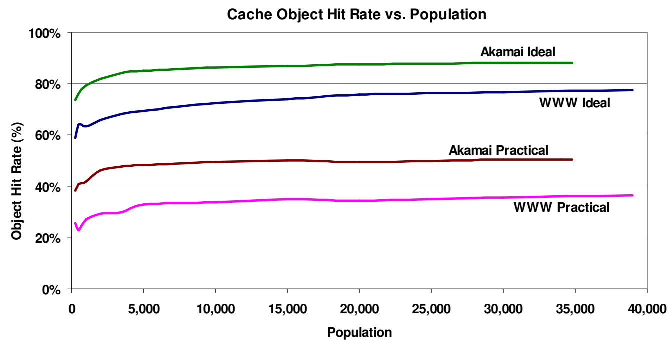


Figure 13. WWW and Akamai object hit rates vs. population: object hit rates as a function of population size. An ideal cache treats all documents as cacheable. A practical cache accounts for HTTP 1.1 caching directives. All simulations used an infinite-capacity proxy cache.

6.1 CDN Caching

Content-delivery networks such as Akamai provide a number of benefits beyond end-user performance, including load balancing, data availability, and reduction of server load. Nonetheless, one interesting question is whether they provide any performance advantage relative to a local proxy cache in an environment such as ours. To answer this question, we simulated the behavior of a UW-local proxy cache against our traced WWW and Akamai workloads.

Figure 13 shows the performance of an infinite-capacity proxy cache for the HTTP transactions in our trace that were sent to Akamai and WWW servers. For both systems, we simulated an "ideal" hit rate (assuming all documents are cacheable) and the hit rate of a "practical" cache. A practical cache accounts for HTTP 1.1 cache control headers [16], cookies, object names with suffixes naming dynamic objects, no-cache pragmas, and uncacheable methods and response codes. For WWW, both the ideal and practical object hit rate curves look similar to the ones from our 1999 study

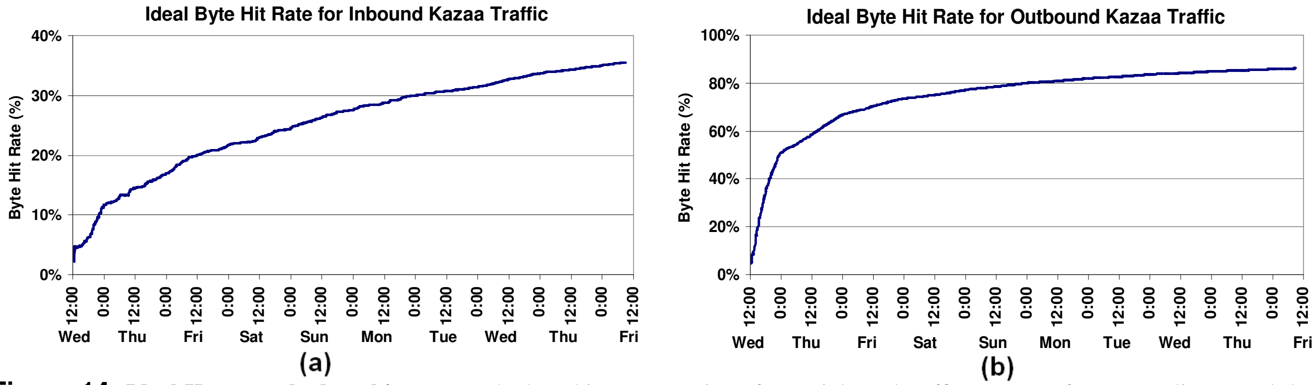


Figure 14. Ideal Kazaa cache byte hit rate: cache byte hit rate over time, for (a) inbound traffic (requests from UW clients) and (b) outbound traffic (requests from external clients).

of the same population [35]. In contrast, the Akamai requests achieve an 88% ideal hit rate and a 50% practical hit rate, noticeably higher than WWW requests (77% and 36% respectively). Our analysis shows that Akamai requests are more skewed towards the most popular documents than are WWW requests, i.e., they are less heavy-tailed, with a Zipf parameter of 1.08 vs. 0.7 for the WWW.

The ideal hit rate for Akamai traffic is extremely high. While many Akamai objects are marked as uncacheable (reflected in the low hit rate of a cache respecting caching directives), we know that most bytes fetched from Akamai are from images and videos; this implies that much of Akamai's content is in fact static and could be cached.

Overall, this analysis indicates that a local web proxy cache could achieve nearly the same hit rate as an Akamai replica (which presumably has a hit rate of 100%), considering in particular that the misses in our proxy are primarily cold-cache effects. Therefore, we would expect that widely-deployed proxy caches would significantly reduce the need for a separate content delivery network, at least with respect to local performance, assuming that immutable objects could be marked as cacheable with long time-to-live values (TTLs).

6.2 P2P Caching

Given the P2P traffic volume that we observed, the potential impact of caching in P2P systems may exceed the benefits seen in the web. In this section, we present an initial exploration of P2P caching. Our goal is not to solve (or even identify) all of the complexities of P2P caching – a full caching study would be outside of the scope of this paper – but rather to gain insight into how important a role caching may play.

To explore this question, we built an ideal (i.e., infinite capacity and no expiration) cache simulator for Kazaa P2P traffic. Since the average P2P object is three orders of magnitude larger than the average web object, we evaluate the benefits of a P2P cache with respect to its byte hit rate, rather than its object hit rate. Because Kazaa peers can transfer partial fragments as well as entire objects, our cache stores items at the granularity of 32 KB *blocks*. For each Kazaa transfer, we identified and cached all complete 32 KB (aligned) blocks. Future requests for the same object may be partially satisfied from these cached blocks. Because of this, a single Kazaa transfer may involve multiple cache block hits and misses.

Figures 14a and 14b show the byte hit rate over time for inbound and outbound Kazaa traffic, respectively. The outbound hit rate (Figure 14b) increases as the cache warms, stabilizing at approximately 85%. This is a remarkably high hit rate – double that reported for web traffic. A reverse P2P cache deployed in the University's ISP would result in a peak bandwidth savings of over 120 megabits per second!

The inbound hit rate grows more slowly over time, reaching only 35% by the end of the trace. It is clear that the simulated inbound cache has not fully warmed even for nine days worth of traffic. Accordingly, we will comment only on outbound traffic for the remainder of this section.

We investigated the source of the high outbound hit rate by examining the 300 top bandwidth-consuming objects. These objects had an average size of 614 MB and a total size of 180 GB. Requests for these 300 objects consumed approximately 5.635 TB of traffic throughout the trace, which is 42% of the total bytes consumed by Kazaa outbound traffic. A conservative estimate suggests that a cache serving only these 300 objects would see a byte hit rate of more than 38% when presented with our entire trace. Thus, a small number of large objects are the largest contributors to the outbound P2P cache byte hit rate.

In Figure 15, we show the cache byte hit rate as a function of population size for outbound traffic. A population of 1,000 clients sees a hit rate of 40%; hit rate climbs slowly thereafter, until a population of 500,000 clients sees a hit rate of 85%. This indicates that a P2P cache would be effective for a small population, but even more effective for large populations.

Currently, many organizations are either filtering or rate-limiting P2P traffic to control bandwidth consumption. Based on our preliminary investigation in this section, we believe caching would have a large effect on a wide-scale

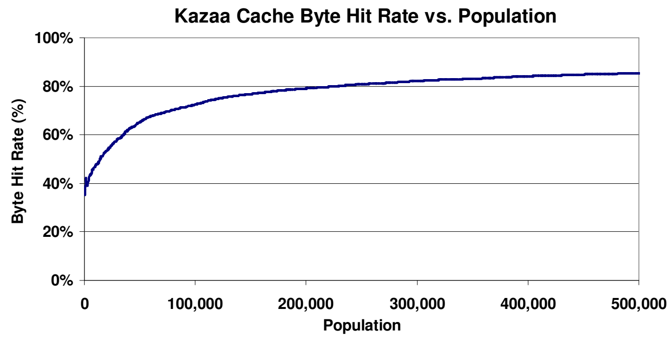


Figure 15. Kazaa cache byte hit rate vs. population: byte hit rate as a function of population size, for outbound traffic.

P2P system, potentially reducing wide-area bandwidth demands dramatically.

7 Conclusions

This paper examined Internet content delivery systems from the perspective of traffic flowing in and out of the University of Washington. To do this, we gathered a trace of all incoming and outgoing traffic over a period of nine days, separating out the HTTP traffic components due to standard web traffic, Akamai-supplied content, Gnutella P2P file transfers, and Kazaa P2P file transfers. Our results confirm that a dramatic shift in Internet traffic and usage has occurred in only several years. Specifically, for our environment we found that:

- Peer-to-peer traffic now accounts for the majority of HTTP bytes transferred, exceeding traffic due to WWW accesses by nearly a factor of three. As a fraction of all TCP traffic, Kazaa alone accounts for 36.9% of bytes transferred, compared to 14.3% for web documents.

- P2P documents are three orders of magnitude larger than web objects, leading to a 1000-fold increase in transfer time. As a result, the number of concurrent P2P flows through the University is approximately twice the number of flows for our web population, despite the extremely low request rate and small population of P2P systems.

- A small number of extremely large objects account for an enormous fraction of observed P2P traffic. For Kazaa transfers out of the University, the top 300 objects, with a total size of 180 GB, were responsible for 5.64 TB of the traffic – almost half of the total outbound Kazaa bandwidth.

- A small number of clients and servers are responsible for the majority of the traffic we saw in the P2P systems. The top 200 of 4,644 UW Kazaa clients accounted for 50% of downloaded Kazaa bytes. More surprisingly, only 600 UW-external peers (out of the 281,026 servers used) provided 26% of the bytes transferred into the University.

- Each P2P client creates a significant bandwidth load in *both* directions, with uploads exceeding downloads for Kazaa users. Our 4,644 Kazaa peers provided 13.573 TB of data to 611,005 external peers, while requesting 1.78 TB of data from 281,026 peers. Overall, the bandwidth requirement of a single Kazaa peer is ninety-fold that of a single web client.

Overall, these points indicate that despite the scalability-based design, the bandwidth demands of peer-to-peer systems such as Kazaa will likely prevent them from scaling further, at least within University-like environments. However, our initial analysis also shows that an organizational P2P proxy cache has the potential to significantly reduce P2P bandwidth requirements. We intend to examine caching in more depth in our future work.

Acknowledgements

We would like to thank Brian Youngstrom who helped us immensely to make our traces a reality. We would also like to thank David Richardson, Art Dong and the other members of the Computing and Communications organization at UW for their support. We are grateful for the guidance of David Culler, our shepherd, and our anonymous reviewers. We thank Alec Wolman and Geoffrey Voelker for their help with the tracing infrastructure. This research was supported in part by NSF grants CCR-0121341, ITR-0085670, and IIS-0205635.

References

[1] Akamai. `http://www.akamai.com`.

[2] J. Almeida, V. Almeida, and D. Yates. Measuring the behavior of a world-wide web server. Technical Report 1996-025, Boston University, Oct. 1996.

[3] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: Evidence and implications. In *Proc. of IEEE INFOCOM 1999*, March 1999.

[4] R. Caceres, F. Douglis, A. Feldmann, G. Glass, and M. Rabinovich. Web proxy caching: The devil is in the details. In *Workshop on Internet Server Performance*, June 1998.

[5] P. Cao, J. Zhang, and K. Beach. Active cache: Caching dynamic contents on the web. In *Proc. of IFIP Int. Conf. on Distributed Systems Platforms and Open Distributed Processing*, Sep. 1998.

[6] S. Chakrabarti, B.E. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the Web's link structure. *Computer*, 32(8), 1999.

[7] A. Chankhunthod, P. B. Danzig, C. Neerdaels, M. F. Schwartz, and K. J. Worrell. A hierarchical internet object cache. In *Proc. of the 1996 USENIX Annual Technical Conf.*, Jan. 1996.

[8] M. Chesire, A. Wolman, G. Voelker, and H. Levy. Measurement and analysis of a streaming media workload. In *Proc. of the 2001 USENIX Symp. on Internet Technologies and Systems*, March 2001.

[9] I. Clarke, O. Sandberg, B. Wiley, and T. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proc. of the ICSI Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[10] Clip2. The Gnutella protocol specification v.0.4, March 2001. `http://www.clip2.com/GnutellaProtocol04.pdf`.

[11] F. Douglis, A. Feldmann, B. Krishnamurthy, and J. C. Mogul. Rate of change and other metrics: a live study of the world wide web. In *Proc. of the 1997 USENIX Symp. on Internet Technologies and Systems*, Dec. 1997.

[12] B. Duska, D. Marwood, and M. J. Feeley. The measured access characteristics of World Wide Web client proxy caches. In *Proc. of the 1st USENIX Symp. on Internet Technologies and Systems*, Dec. 1997.

[13] L. Fan, P. Cao, J. Almeida, and A. Z. Broder. Summary cache: A scalable wide-area web cache sharing protocol. In *Proc. of ACM SIGCOMM '98*, Aug. 1998.

[14] S. Gadde, J. Chase, and M. Rabinovich. Web caching and content distribution: A view from the interior. In *Proc. of the 5th International Web Caching and Content Delivery Workshop*, May 2000.

[15] S. D. Gribble and E. A. Brewer. System design issues for internet middleware services: Deductions from a large client trace. In *Proc. of the 1997 USENIX Symp. on Internet Technologies and Systems*, Dec. 1997.

[16] Internet Engineering Task Force. Hypertext transfer protocol - http 1.1. RFC 2068, March 1997.

[17] Internet2. `http://netflow.internet2.edu/weekly/20020422`.

[18] K. L. Johnson, J. F. Carr, M. S. Day, and M. Frans Kaashoek. The measured performance of content distribution networks. *Computer Communications*, 24(2), 2001.

[19] J. Kangasharju, K.W. Ross, and J.W. Roberts. Performance evaluation of redirection schemes in content distribution networks. *Computer Communications*, 24(2):207–214, 2001.

[20] Kazaa. `http://www.kazaa.com`.

[21] J. M. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, and A. S. Tomkins. The Web as a graph: Measurements, models, and methods. In T. Asano, H. Imai, D. T. Lee, S. Nakano, and T. Tokuyama, editors, *Proc. of the 5th Annual Int. Conf. Computing and Combinatorics*, number 1627. Springer-Verlag, 1999.

[22] M. Koletsou and G. M. Voelker. The Medusa proxy: A tool for exploring user-perceived web performance. In *Proc. of the Sixth Int. Workshop on Web Caching and Content Distribution*, June 2001.

[23] B. Krishnamurthy, C. Wills, and Y. Zhang. On the use and performance of content distribution networks. In *Proc. of SIGCOMM IMW 2001*, Nov. 2001.

[24] N. Leibowitz, A. Bergman, R. Ben-Shaul, and A. Shavit. Are file swapping networks cacheable? Characterizing P2P traffic. In *Proc. of the 7th Int. WWW Caching Workshop*, August 2002.

[25] B. Maggs. Global Internet Content Delivery. Talk delivered in the Internet and Distributed Systems Seminar at Stanford University. `http://www.stanford.edu/class/cs548/abstracts.shtml#bruce`.

[26] S. McCanne and V. Jacobson. The BSD packet filter: A new architecture for user-level packet capture. In *Proc. of the USENIX Technical Conf.*, Winter 1993.

[27] J.-M. Menaud, V. Issarny, and M. Banatre. A new protocol for efficient transversal Web caching. In *Proc. of the 12th Int. Symp. on Distributed Computing*, Sep. 1998.

[28] Napster. `http://www.napster.com`.

[29] V. N. Padmanabhan and L. Qiu. The content and access dynamics of a busy web site: Findings and implications. In *Proc. of ACM SIGCOMM 2000*, August 2000.

[30] M. Rabinovich, J. Chase, and S. Gadde. Not all hits are created equal: Cooperative proxy caching over a wide area network. In *Proc. of the 3rd Int. WWW Caching Workshop*, June 1998.

[31] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proc. of Multimedia Computing and Networking 2002*, Jan. 2002.

[32] A. Shaikh, R. Tewari, and M. Agrawal. On the effectiveness of DNS-based server selection. In *Proc. of IEEE INFOCOM 2001*, Anchorage, AK, USA 2001.

[33] R. Tewari, M. Dahlin, H. Vin, and J. Kay. Design considerations for distributed caching on the Internet. In *The 19th IEEE Int. Conf. on Distributed Computing Systems*, May 1999.

[34] D. Wessels, K. Claffy, and H.-W. Braun. NLANR prototype web caching system. `http://ircache.nlanr.net/`.

[35] A. Wolman, G. Voelker, N. Sharma, N. Cardwell, M. Brown, T. Landray, D. Pinnel, A. Karlin, and H. Levy. Organization-based analysis of web-object sharing and caching. In *Proc. of the 2nd USENIX Conf. on Internet Technologies and Systems*, Oct. 1999.

[36] A. Wolman, G. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. Levy. The scale and performance of cooperative web proxy caching. In *Proc. of the 17th ACM Symp. on Operating Systems Principles*, Dec. 1999.

[37] B. Yang and H. Garcia-Molina. Efficient search in peer-to-peer networks. In *Proc. of the 22nd Int. Conf. on Distributed Computing Systems*, July 2002.