Peer-Assisted Content Distribution in Akamai NetSession

Mingchen Zhao[†] Paarijaat Aditya[‡] Ang Chen[†] Yin Lin[⋆◇]
Andreas Haeberlen[†] Peter Druschel[‡] Bruce Maggs[⋆◇] Bill Wishon[◇] Miroslav Ponec[◇]
[†]University of Pennsylvania [‡]MPI-SWS [⋆]Duke University [◇]Akamai Technologies

ABSTRACT

Content distribution systems have traditionally adopted one of two architectures: infrastructure-based content delivery networks (CDNs), in which clients download content from dedicated, centrally managed servers, and peer-to-peer CDNs, in which clients download content from each other. The advantages and disadvantages of each architecture have been studied in great detail. Recently, hybrid, or "peer-assisted", CDNs have emerged, which combine elements from both architectures. The properties of such systems, however, are not as well understood.

In this paper, we discuss the potential risks and benefits of peer-assisted CDNs, and we study one specific instance, Akamai's NetSession system, to examine the impact of these risks and benefits in practice. NetSession is a mature system that has been operating commercially since 2010 and currently has more than 25 million users in 239 countries and territories. Our results show that NetSession can deliver several of the key benefits of both infrastructure-based and peer-to-peer CDNs—for instance, it can offload 70–80% of the traffic to the peers without a corresponding loss of performance or reliability—and that the risks can be managed well. This suggests that hybrid designs may be an attractive option for future CDNs.

Categories and Subject Descriptors

C.2.4 [**Computer Systems Organization**]: Computer-Communication Networks—*Distributed Systems*; C.4 [**Computer Systems Organization**]: Performance of Systems

Keywords

Content distribution networks, peer-to-peer systems

1. INTRODUCTION

This paper presents a study of a hybrid CDN called NetSession in which most content is delivered by a set of peers whose operation is coordinated (and backstopped) by a dedicated infrastructure. Our study is motivated by the observation that hybrid CDNs seem to have reconciled two architectures with very different tradeoffs: On the one hand, peer-to-peer systems are inexpensive and easy to scale but seem to be plagued by security issues and low quality of service (QoS); on the other hand, infrastructure-based systems are expensive to set up and scale but can provide predictable QoS and reliable accounting, and ensure content integrity. We felt that the time was ripe to investigate what advantages a hybrid of these approaches might convey.

We find, perhaps surprisingly, that NetSession is able to achieve the "best of both worlds": it can offer most of the benefits of both architectures while avoiding most of the drawbacks. This is because the strengths of the infrastructure and the peers complement each other: The infrastructure provides a central point of coordination that can quickly match up peers and can add resources when peers cannot provide adequate QoS; the peers provide resources and scalability, and they extend the "reach" of the infrastructure to underserved areas.

Prior work has identified several potential concerns about hybrid CDNs; in particular, it has been suggested that hybrid CDNs might become a burden to Internet Service Providers (ISPs) by increasing inter-AS traffic [17, 24, 35]. We show that, at least in the case of NetSession, this concern appears to be unfounded. Finally, we report some observations from the day-to-day operation of the NetSession system. Among other things, we discover a surprising degree of user mobility in the system, and we describe what appear to be the effects of cloning and re-imaging client installations. In summary, this paper makes the following five contributions:

- an overview of the hybrid CDN design space, including a discussion of potential risks and benefits (Section 2);
- a description of NetSession's design goals and implementation (Section 3);
- a measurement study that provides an overview of the scale at which NetSession operates and the types of content it is used to deliver (Section 4);
- an analysis of whether NetSession meets its design goals and realizes the potential benefits of hybrid systems (Section 5); and
- an investigation into whether the risks associated with hybrid systems, such as undue burdens on ISPs or problems with untrusted end-user machines, are evidenced by NetSession in practice (Section 6).

We discuss related work in Section 7, and present our conclusions in Section 8.

IMC'13, October 23–25, 2013, Barcelona, Spain.
ACM 978-1-4503-1953-9/13/10. http://dx.doi.org/10.1145/2504730.2504752.

2. THE CDN DESIGN SPACE

We begin by sketching the design space for content distribution systems, and discuss the potential risks and benefits of hybrid designs.

2.1 To Peer or Not To Peer?

Traditionally, CDNs have fallen into two categories: some relied exclusively on a centrally managed infrastructure, while others operated entirely without one. Akamai's CDN [10] is an example of an *infrastructure-based* CDN: it uses an infrastructure of more than 137,000 servers in 87 countries within over 1,150 Internet networks, all owned and operated by Akamai. BitTorrent [8] is an example of a *peer-to-peer (p2p)* CDN: other than the tracker (which serves as an initial point of contact) and perhaps an initial seeder, it requires no infrastructure at all; all data is exchanged directly between the peers. Recent BitTorrent clients have even replaced the tracker with a distributed hashtable [9], removing the last remaining infrastructure element.

Infrastructure-based CDNs typically have the benefit of professional administrators and amply provisioned resources—they can control and authenticate the content they distribute, and they can achieve high performance and reliability, but they are also expensive to scale. Peer-to-peer CDNs must rely on resources contributed by their peers, which means that their properties are often the exact opposite: they require little up-front investment and scale organically, but can be plagued by spam and low-quality content, and their quality of service tends to be unpredictable. Thus, it is natural to ask whether it might not be possible to achieve the "best of both worlds" in a single system.

2.2 Why decide?

In this paper, we focus on hybrid CDNs that combine elements from both architectures: like peer-to-peer CDNs, they achieve scalability by leveraging resources from the client machines, but, like infrastructure-based CDNs, they also rely on dedicated, centrally managed elements, such as a central directory of file locations and dedicated servers that provide a backstop of delivery capacity. We refer to such CDNs as *peer-assisted CDNs*.

One way to characterize peer-assisted CDN designs is to look at the role they assign to the infrastructure. For instance, the infrastructure can provide *resources*, such as bandwidth, computation, or storage, which might be used to improve the quality of service or to keep rarely requested content available; it can provide *coordination* by building a global view of the system, e.g., to recognize attacks, to allocate resources, or to pair up peers with matching NATs; and it can provide *control*, e.g., by deciding which content may be distributed. Thus, there is an entire spectrum of possible designs, ranging from systems that rely mostly on the peers and have a very small infrastructure (such as tracker-based BitTorrent) to systems that rely primarily on the infrastructure and use the peers only as an optimization.

2.3 Potential benefits

Peer-assisted CDNs have substantial flexibility in dividing tasks between the peers and the infrastructure. Thus, they can potentially use each element where it is strongest. The potential benefits relative to a pure peer-to-peer CDN include:

Better quality of service: The infrastructure can be used as a "backstop" for the peers, i.e., it can help out with requests where the peers do not have much bandwidth, or where there is a lot of churn.

Reduced reliance on peer contributions: The infrastructure can absorb the cost of a certain degree of freeloading. Thus, it is not as important to force peers to reciprocate, or to "fairly" distribute the workload.

More reliable delivery: The infrastructure can authenticate content; there is no need to rely on heuristics or reputations to recognize content that has been corrupted or tampered with.

Less legal exposure: The infrastructure can ensure that all content is properly licensed; thus, peers can safely participate in swarming and upload content to others, without being at risk for accidental violations of the DMCA (and similar laws).

Better security: The infrastructure can use its global view to recognize and fight attacks, and it can provide professionally managed servers that can perform trusted functions.

Higher efficiency: The infrastructure can maintain a global view and perform global optimizations, e.g., by quickly locating copies of a file, or by matching up peers.

The potential benefits over a pure infrastructure-based system include:

Lower cost: Content providers can potentially deliver their data at lower cost (and can pass some of the resulting savings on to end users). Also, the CDN operator may be able to offer services to content providers who otherwise may not be able to afford a CDN.

Global coverage: Finally, in geographic areas with sparse infrastructure deployment, serving content from peers may increase the quality of service. Also, ISPs that are not hosting CDN servers should see less inter-ISP traffic for popular downloads.

2.4 Potential risks

Peer-assisted CDNs face some of the risks that affect pure infrastructure based and peer-to-peer architectures, and a few risks that are unique to them:

Need for revenue: Unlike peer-to-peer CDNs, peer-assisted CDNs must generate some revenue to defray the costs of the infrastructure.

Less transparency: Unlike infrastructure CDNs, peer-assisted CDNs are usually not transparent to users; they require the installation of software on each client.

Heterogeneity: Unlike most infrastructure CDNs, peer-assisted CDNs include a heterogeneous population of machines that are managed by users, with varying capabilities and complex failure modes.

Multiple administrative domains: Unlike infrastructure CDNs, peer-assisted CDNs inherently contain machines that are controlled by different parties, many of whom the CDN operator does not trust.

NATs and firewalls: Unlike infrastructure CDNs, peer-assisted CDNs rely on their clients' upstream links, which requires traversing a variety of middleboxes that may exist on the path.

Impact on ISPs: Peers in peer-assisted CDNs send *and* receive traffic; thus, these systems are creating a different traffic pattern, which can affect ISPs' networks and businesses.

3. THE NETSESSION SYSTEM

To examine how many of the potential benefits peer-assisted CDNs can deliver in practice, and how many of their potential risks they can avoid, we have studied the NetSession system, a peer-assisted CDN that was originally developed by RedSwoosh and has been commercially operated by Akamai since 2010. As of October 2012, NetSession has been in production use for five years and has almost 26 million users in 239 countries and territories.

3.1 Design goals

NetSession was designed with the following three high-level goals in mind:

1. A substantial fraction of the content should be delivered by the peers.
2. Peer-assisted delivery QoS should be comparable to that of infrastructure-based delivery; in particular,
   (a) downloads should be no less reliable; and
   (b) downloads should not be much slower.
3. The system should offer reliable accounting for services provided.

In other words, the system was meant to combine the key benefits of peer-to-peer CDNs (scalability) and infrastructure CDNs (quality of service). The third goal was an operational requirement: Content providers, who pay for the CDN's services, expect detailed logs that show the amount and the quality of the services provided. There were also two explicit non-goals:

1. The system need not be *more* reliable than an infrastructure-based CDN; and
2. Peers need not contribute equally.

The first point sets realistic expectations about security; the second point reflects the fact that the system has a large infrastructure to fall back on, so some proportion of peers who opt out of serving content to peers would not be a concern. Serving content reliably and with good QoS is more important than minimizing load on the infrastructure.

3.2 Architecture

NetSession distributes content via an infrastructure of *edge servers* that are operated by Akamai, and a number of user-operated peers that have special software, the *NetSession Interface* (Section 3.4), installed on them. In addition to the edge servers, the infrastructure also contains a group of NetSession-specific servers called the *NetSession control plane* (Section 3.6), which serve as coordinators and perform accounting, but do not directly serve any content. Figure 1 illustrates the high-level interaction between these components.

To avoid confusion, we use the term *content providers* to refer to the organizations and individuals who provide content for NetSession to distribute, and the term *users* to refer to those who download the content.

3.3 Example: Download Manager

We begin by briefly describing an example application, the Download Manager (DLM). The DLM is one of several applications that use the NetSession system; a typical use case

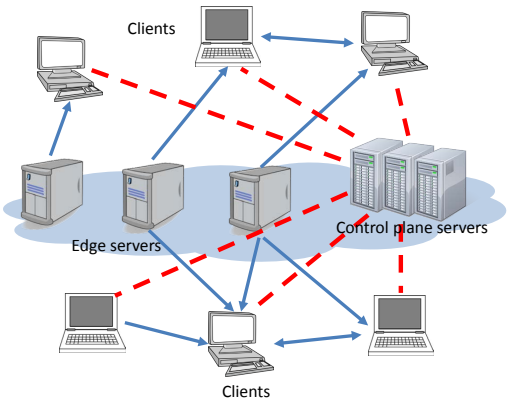


Figure 1: Overview of the NetSession system.

is to distribute large objects that are several GBs in size, such as software installation images.

When a user attempts to download an object that is distributed using the DLM, she is first asked to install the NetSession Interface on her computer (unless it is already available from previous usage). Once installed, the NetSession Interface starts downloading the content from the edge servers; in parallel, it queries the control plane for a list of nearby peers that already have a copy of the object. If suitable peers are found, the local peer and the selected peers attempt to contact each other and exchange as much data as possible; however, the download from the edge servers continues in parallel. Thus, if a peer is "unlucky" and picks peers that are slow or unreliable, the infrastructure can cover the difference, so that user experience does not suffer as a result.

Users can pause and resume downloads, and they can continue downloads that were aborted earlier, e.g., because the peer lost network connectivity or the peer's hard drive was full. Once the download completes, the NetSession Interface software remains on the peer and can be reused for future downloads, or by other NetSession-enabled applications, and it can upload the downloaded content to other peers.

3.4 The NetSession Interface

The NetSession Interface is available for Windows and Mac OS. It is implemented as a background application that runs whenever the user is logged into their system. This design choice is different from many p2p clients, which must be launched explicitly by the user. The short session times that have been observed in p2p systems [4, 14, 27] suggest that users launch the client only when they intend to download something, so the time window in which objects can be uploaded to other peers tends to be very short. As a persistent background application, NetSession does not have this problem, but in return, it must take great care not to inconvenience the user. We discuss some of its best practices in Section 3.9.

Whenever the NetSession Interface is active and the peer is online, it maintains a TCP connection to the control plane. When a download is started, the peer uses this control connection to query the control plane for other peers. The connection is also useful for opening peer-to-peer connections through NATs, which in most cases requires coordination between the peers; the control plane can facilitate this by informing both endpoints using their control connection. Fi-

nally, peers use the connection to learn about configuration updates, and they report usage statistics, which are used for billing, performance monitoring, and to generate reports for content providers. Each peer has a unique GUID, which is chosen at random during installation.

NetSession uses the standard HTTP (or HTTPS) protocol to download content from edge servers; for downloads from peers, it uses a swarming protocol not unlike BitTorrent's. As in BitTorrent, objects are broken into fixed-size pieces that can be downloaded and their content hashes verified separately, and peers exchange information about which pieces of the file they have locally available.

A key difference to BitTorrent is the absence of an incentive mechanism: in NetSession, peers can always obtain the content from the infrastructure, so it is not as important to discourage "freeloading"—the infrastructure can easily absorb the cost of a few users who decide not to upload. Hence, a conscious decision was made not to include incentives, and to serve content to each peer at the best possible speed, regardless of how much bandwidth it is contributing to the system. There is no tit-for-tat strategy that would "choke" slow uploaders as in BitTorrent, only a globally configurable limit on the total number of upload connections a peer allows. In fact, NetSession Interface users have the option to turn off peer content uploads permanently or temporarily in the NetSession application preferences, without adverse effects on their download performance.

NetSession is predominantly used by content providers who distribute large objects and already require their clients to install software (e.g., games). NetSession also supports video streaming, but it currently does not serve much video traffic because of the requirement to install client software.

3.5 Interaction with edge servers

NetSession's HTTP(S) connections to the edge servers are used not only for downloading files, they also support many other critical functions.

One important function is to ensure content integrity. File pieces can be corrupted in transit or on the peers; additionally, content can change over time, so it is important that different versions are not mixed up in the same download. Edge servers generate and maintain secure IDs of content, which are unique to each version, as well as secure hashes of the pieces of each file. The IDs and the hashes are provided to the peers, so they can validate the content they have downloaded. If a peer cannot validate a file piece, it discards the piece and does not upload it to other peers.

Another key function is authorization. Before a peer can receive content from other peers, it must authenticate to an edge server over the HTTP(S) connection; this yields an encrypted token that can be used to search for peers. This is done to prevent users from downloading files from peers that they are not authorized to obtain from the infrastructure.

Finally, the HTTP(S) connections are used for configuration and reporting. A policy defined by the content provider is used to decide whether a particular file may be downloaded and uploaded; in addition, various configurable options apply to each download and upload. These policies and options are securely communicated to the peers through the trusted edge-server infrastructure. NetSession also uses information from the trusted edge servers to prevent accounting attacks [1], where compromised or faulty peers incorrectly report downloads and uploads.

3.6 The NetSession control plane

The NetSession control plane consists of a number of globally-distributed servers that are operated by Akamai. Its main function is to coordinate between the peers. Each control plane server runs some of the following components:

Connection node (CN): The CNs are the endpoints of the persistent TCP connections that the peers open to the control plane when they are active. The CNs receive and collect the usage statistics that are uploaded by the peers, and they handle queries for objects the peers wish to download. These persistent TCP connections are also used to tell peers to connect to each other in order to facilitate sharing of content. Such coordination is necessary for both security reasons and to overcome NATs and firewalls.

Database node (DN): The DNs maintain a database of which objects are currently available on which peers, as well as details about the connectivity of these peers. Peers appear in the database only when a) uploads are explicitly enabled on the peer, and b) the peer currently has objects to share.

STUN: Peers periodically communicate with STUN components over UDP and TCP to determine the details of their connectivity (which are then stored in the DN databases) and to enable NAT traversal. This involves a protocol with goals similar to [25], but NetSession uses a custom implementation.

Monitoring node: Peers upload information about their operation and about problems, such as application crash reports, to these nodes. Processing their logs helps to monitor the network in real-time, to identify problems, and to troubleshoot specific user issues during support procedures.

3.7 Peer selection

When a CN receives a query for an object with peer-to-peer delivery enabled, the CN asks the DNs to identify suitable peers that currently have a copy of the requested object. The CN then returns information about these peers to the querying peer, and it instructs both the querying peer and the chosen peers to initiate connections to each other. By default, up to 40 peers are returned, and if connections to some of these peers cannot be established, additional queries are issued until a sufficient number of peer connections succeed. Peers control the number and utilization of their connections based on current resource availability.

The DN chooses peers using a locality-aware strategy at two different levels. First, when a peer establishes its persistent TCP connection to the control plane, it is mapped to the closest available CN by Akamai's DNS system [10]. When a CN queries the DN for peers for a specific object, it prefers to contact only local DNs, i.e., DNs running on machines in the same network region as the CN that performs the query. (Network regions are defined by proximity to particular groups of servers.) Since the same process is used when a peer registers a local copy of a file, DNs tend to have information about their *local* peers. The CN/DN system is interconnected across regions, so it is possible in principle to search for peers from any region; however, long-term experimentation has shown that using only local DNs in searches does not negatively impact performance.

Since the current deployment has less than 20 network regions, the first, region-based selection strategy is not sufficiently fine-grained for popular content that is available on many peers. Hence, the DNs use another level of locality-

based peer selection that is based on the geolocation of each peer. Each peer belongs to multiple sets, based on its public IP address and the Autonomous System (AS) it is located in. For example, a peer can simultaneously be in a universal *World* set, a subset for a large geographical region, a subset for a smaller region, and a subset for its specific AS.

DN selection begins with peers from the most specific set that the querying peer belongs to, and proceeds to less specific sets until enough suitable peers are found. An additional mechanism adds diversity: Occasionally, peers are selected from a less specific set, with probability proportional to the specificity of the set. Also, when a peer is selected, it is placed at the end of a peer selection list for fairness. The selection process can be modified with a set of configurable policies.

In addition to locality and file availability, the DN also takes the connectivity of the peers into account: it selects only peers that are likely to be able to establish a connection with each other, e.g., based on the type of their NAT or firewall. Due to the vast diversity in NAT implementations today, NAT hole punching is a complex issue, and the necessary code takes up a large fraction of the NetSession codebase.

3.8 Robustness

The design of NetSession employs the notions of soft state and fate sharing to provide robustness against failures. At first glance, it might seem that the loss of CN or DN components could be catastrophic to NetSession. Indeed, many peers rely on each CN: over 150,000 might be connected to one simultaneously. But ultimately, all of the data about the peers that matters is held by the peers themselves. If a CN goes down, the peers that are connected to that CN simply reconnect to another one. If a DN goes down, the CNs connected to that DN send a RE-ADD message to their peers, asking them to list the files that they are storing. The CN passes these lists on to the available DNs in order to repopulate their databases. In practice, failures of CN and DN nodes occur routinely, e.g., during server maintenance or during software updates. In fact, when a new CN/DN software version is released, all CNs and DNs are restarted in a short timeframe, and this does not negatively affect the service. (In the event of an unexpectedly large-scale failure, reconnections are rate-limited to ensure a smooth recovery.) Finally, if a peer is not able to connect to any CN at all, it retrieves the content directly from the edge servers; hence, even if the entire CN and DN infrastructure were to fail, the peers would simply fall to back to retrieving content from the CDN infrastructure.

The client software version is centrally controlled by the CDN infrastructure, and peers can perform automated upgrades in the background on demand. Most of the peer population can be upgraded to a new version within one hour. The ability to perform fast software upgrades without user interaction can help to respond quickly to security or performance incidents. Download and upload performance is constantly monitored, and automated alerts are in place to notify network engineers in case of large-scale problems.

3.9 Best practices

Since NetSession uses resources that are provided by the peers, it must carefully consider the users' interests. NetSession obtains consent from users through its EULA, and Akamai provides users with information about what the NetSession Interface is, and what it does. The software includes both a control panel user interface and a command line utility that enable users to determine what the software is doing, which files it is currently storing, which applications are using it, etc. These tools also allow users to turn uploading on or off, and it comes with an uninstaller.

NetSession is designed to protect the privacy of its users; it does not capture personal information [3]. The data it keeps is similar to that of a normal web server. When peers download content from each other, they learn that other peers with certain GUIDs and IP addresses already have the requested file. However, this information is not displayed to users and is deleted from the peer once a download completes and the logs have been uploaded to the CNs for billing and monitoring.

To avoid inconveniencing users, the NetSession Interface is designed to stay in the background as much as possible. For example, a peer does not proactively download content; it only shares objects that the corresponding user has previously downloaded. Uploads are rate-limited, and peers upload each object at most a limited number of times. Finally, peers monitor the utilization of the local network connections and throttle or pause uploads when the connections are used by other applications. While it is important for users to experience good download performance, the performance of uploads is intentionally limited using custom protocols.

Users do not benefit directly from the bandwidth they donate, but NetSession offers a number of indirect benefits; for instance, the DLM enables users to resume aborted downloads and to download from multiple sources simultaneously. Also, peer-assisted downloads require fewer resources from the infrastructure and can thus be offered at a lower price. Content providers can pass on these savings to the users.

Control plane logs:	
Time period covered	10/01 – 10/31, 2012
Log entries	4,150,989,257
Number of GUIDs	25,941,122
Control plane servers	197
Distinct URLs	4,038,894
Distinct IPs	133,690,372
Downloads initiated	12,508,764
Geolocation data:	
Distinct IPs	133,690,372
Distinct locations	34,383
Distinct autonomous systems	31,190
Distinct country codes	239

Table 1: Overall statistics for our data sets.

4. MEASUREMENT STUDY

We now turn to the question how well NetSession, as one instance of a peer-assisted CDN, is able to deliver the potential benefits of a hybrid architecture, and how well it is able to avoid the corresponding risks. To answer this question, we have performed a measurement study that is based on a set of logs from the production system.

4.1 Data set

Our logs cover the month of October 2012. At a high level, they contain information about *downloads* and information about *logins*. When a peer downloads a file from NetSession,

	Americas			Asia			Europe	Africa	Oceania
	US East	US West	Other	India	China	Other			
Customer A	–	–	12%	6%	6%	18%	51%	4%	3%
Customer B	2%	1%	1%	11%	–	61%	6%	17%	1%
Customer C	13%	6%	15%	1%	–	8%	55%	1%	2%
Customer D	22%	21%	6%	–	–	3%	45%	–	3%
Customer E	5%	3%	8%	2%	1%	29%	48%	2%	3%
Customer F	–	–	–	–	–	–	100%	–	–
Customer G	8%	3%	12%	2%	8%	20%	45%	2%	2%
Customer H	6%	4%	7%	4%	2%	20%	53%	2%	2%
Customer I	5%	2%	18%	–	–	15%	57%	1%	1%
Customer J	42%	24%	14%	–	–	5%	11%	1%	3%
All customers	7%	4%	11%	3%	2%	20%	46%	4%	2%

Table 2: Global distribution of downloads for the ten largest content providers.

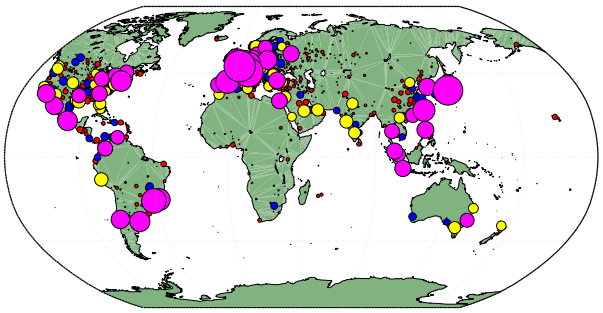


Figure 2: Global distribution of peers.

the CN records information about the download, including the GUID of the peer, the name and size of the file, the CP code (a number identifying a specific account of a content provider that is offering the file), the time the download started and ended, and the number of bytes downloaded from the infrastructure and from peers. This information is used for accounting and billing. Additionally, when a peer opens a connection to the control plane, the CN records the peer's current IP address, its software version, and whether or not uploads are enabled on that peer.

To localize the peers geographically and in the network, we also obtained geolocation data from Akamai's EdgeScape [2] service about each IP address that appears in the trace. This data includes an ISO 3166 country code, the name of a city and state, a latitude/longitude pair, a timezone, and a network provider name. (Note that territories and areas of geographical interest, such as Antarctica, can also have country codes.) The granularity of the location information varies by region; in the United States, locations are typically at the city/suburb granularity. For instance, the data set contains 218 unique locations in the state of Pennsylvania.

To protect the privacy of users and content providers, the data in our logs have been anonymized by hashing the file names, IP addresses, and GUIDs. Table 1 shows some overall statistics for our data set.

4.2 Number and location of the peers

We begin by giving an overview of the NetSession deployment as of October 2012, to illustrate the number and geographic distribution of the peers, as well as the type of content being served. The one-month trace contains about 26 million distinct GUIDs. (Recall that the NetSession software chooses a GUID when it is first installed, so the number of GUIDs should correspond roughly to the number of peers.) On a typical day, between 8.75 and 10.90 million of the GUIDs connect to the control plane at least once. The system has been growing steadily over time; for comparison, a trace from October 2010 contained 14.19 million distinct GUIDs, or slightly more than half the number in our trace.

Figure 2 shows the global distribution of the peers as a "bubble plot": the size of each bubble corresponds to the number of peers whose first connection to the system was from that particular location. Most of the peers are located in North America (27%) and Europe (35%), but there are also sizable groups of peers in South America and Asia. Overall, we observed connections from 239 different countries and territories across all continents, so NetSession is a truly global system.

4.3 Content providers

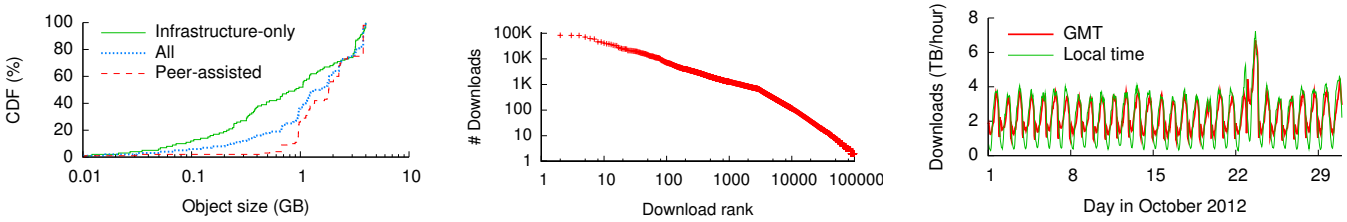
Each file that is available through NetSession is offered by a specific content provider—usually a large corporate customer. To illustrate how these customers are using the system, we selected the downloads served by the ten largest customers (here identified as Customers A through J), we associated each download with one of ten regions, and we counted the number of downloads per customer per region.

Table 2 show the results. Generally, we find that roughly half of the downloads occur in Europe. However, the downloads distribution depends on customer. Customer B's content, for example, is far more popular in Asia except China and Africa. Customer J's content is mostly requested from within the United States.

4.4 Available content

At the time our data was collected, a typical use case for NetSession was the distribution of software installers; data files and other content, such as music and video, made up a small portion. Overall, we observed downloads for 4,038,894 distinct objects in our trace. Figure 3(a) shows the distribution of requests for objects of a given size, for peer-assisted, infrastructure only, and all requests in NetSession. Peer-assisted downloads are strongly biased towards large files; 82% of peer-assisted requests are for objects larger than 500 MB. Because the benefits of peer assist are most pronounced for such large objects, content providers tend to enable it on such objects.

The object popularity distribution and temporal request pattern in NetSession's workload are shown in Figure 3(b) and (c). As expected, the former shows the nearly ubiquitous power law, while the latter shows the usual diurnal patterns.



(a) Request distrib. by object size (b) Content popularity (c) Bytes served over time

Figure 3: Overall workload characteristics.

Uploads initially...	Nodes	Number of changes		
		0	1	≥ 2
Disabled	15,913,255	99.96%	0.03%	0.01%
Enabled	7,395,867	98.11%	1.80%	0.09%

Table 3: Observed changes to the setting that enables content uploads.

Customer	A	B	C	D	E	F	G	H	I	J
p2p (%)	<1	20	2	94	2	45	47	<1	91	<1

Table 4: Fraction of peers that have content uploads enabled.

5. BENEFITS

Next, we focus on three questions that can be answered quantitatively based on our data, namely: 1) How well does peer assist work? 2) Does peer assist affect performance and reliability? And, 3) does peer assist help improve the CDN's global coverage?

5.1 How well does peer assist work?

Recall that NetSession peers are not required to contribute bandwidth: users are free to disable uploads to peers by changing their preferences in the NetSession GUI. This is a major difference from pure peer-to-peer systems like BitTorrent, which include incentive mechanisms like tit-for-tat to encourage uploading. In the literature, it is often assumed that users are "rational" and will avoid uploading if they don't benefit from it [8, 23]; it is therefore natural to ask whether NetSession peers are willing to contribute any bandwidth at all.

Do the peers contribute resources? To answer this question, we used the login records in our data set to determine the fraction of peers that have uploads enabled. However, the NetSession binary is available in two versions, one with uploads initially enabled and one with uploads initially disabled. This is chosen by the content provider who bundles the binary; some content providers use the software merely as a download manager, without the peer assist. For this reason, we additionally check whether users changed this setting between logins, and if so, how often.

Tables 3 and 4 show our results. About 31% of the peers have uploading enabled, but the setting is rarely changed—more than 99% of the peers keep their initial setting throughout our trace. As Table 4 shows, the initial setting depends on the content provider from who the user first downloaded the binary. Most users simply stick with whatever the default is. This tendency is well known in UI design [21, 31], but it also suggests that users either don't care enough about the uploads to change the setting or are not aware of the choice, despite its mention in the NetSession user agreement.

Uploading in peer-to-peer CDNs carries the risk of legal exposure and/or degraded network performance [29]. As a peer-assisted CDN, NetSession can avoid the first problem because its content is centrally controlled and vetted, and its back-off mechanism (Section 3.9) can avoid the second problem by consuming bandwidth only when the connection is idle. We speculate that this is part of the reason why most users don't seem to turn off peer uploads when they are enabled initially.

How much can be offloaded to the peers? As a peer-assisted CDN, NetSession can offload some of the bandwidth needed to satisfy the download requests to the peers. Content providers can control on a per-file basis whether or not peer-to-peer downloads are allowed. In our trace, we found that peer-to-peer downloads were enabled for only 1.7% of the files, but these downloads accounted for 57.4% of the downloaded bytes overall.

The key quantity of interest is the *peer efficiency* of the system, i.e., the fraction of bytes in peer-assisted downloads that are actually downloaded from the peers. (The peer efficiency for infrastructure-only downloads is zero.) In our trace, the average peer efficiency for peer-assisted downloads was 71.4%. This is a very good result, given that, even in a peer-assisted download, NetSession never *exclusively* downloads from the peers; there is always at least one connection to the infrastructure, to guarantee progress independent of the peers.

5.2 Does peer assist reduce performance?

Are peer-assisted downloads slower? NetSession relies on peer downloads, which are limited by peer reliability and the upstream bandwidth of broadband access networks, to which many of NetSession's peers are connected. This bandwidth is typically much smaller than the downstream bandwidth [11]. Hence, it is natural to ask how the performance of peer-assisted downloads compares to those that are served by the infrastructure.

Figure 4 makes this comparison for downloads from the two networks with the most downloads, AS X and AS Y. We identified all downloads from these networks where either a) all the bytes came from the edge servers, or b) at least 50% of the bytes came from peers. We then averaged the speed of each download across its entire length; the figure shows the results as a CDF.

We find that, although the peer-assisted downloads are somewhat slower, the speed is still quite high, with most downloads occurring at rates of multiple Mbps. When comparing the performance across networks, we find that the

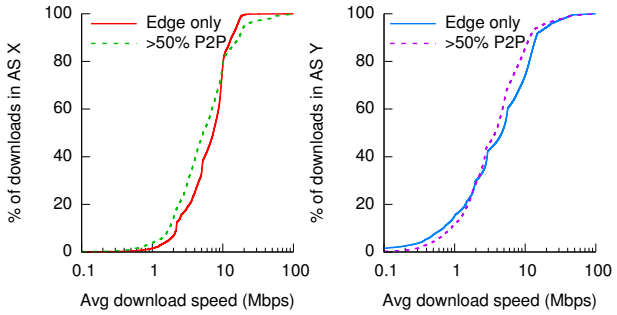


Figure 4: Edge-only vs peer-assisted download speed in two large ASes.

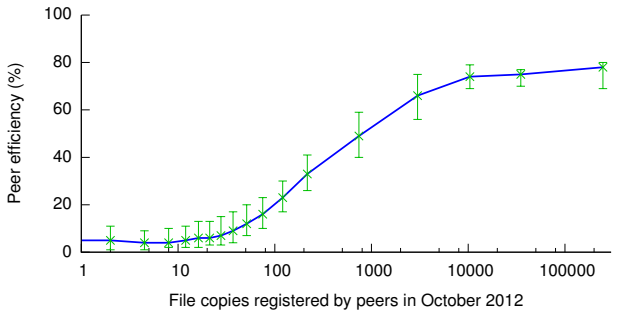


Figure 5: Number of registered file copies vs. peer efficiency

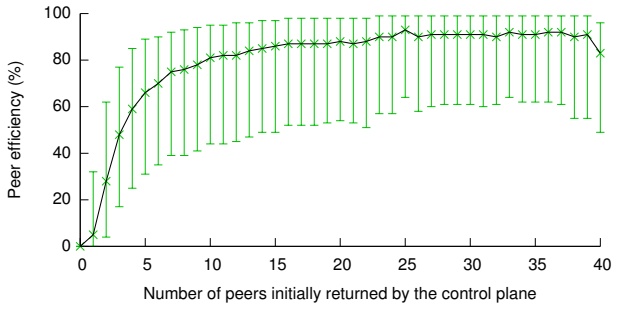


Figure 6: Impact of the number of peers on peer efficiency.

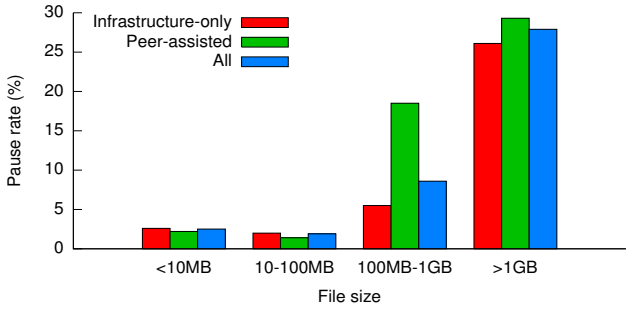


Figure 7: Downloads of larger files are terminated more often.

biggest differences between peer-assisted and infrastructure download speeds occur in the networks with the highest link bandwidths. One possible explanation is the high asymmetry of most high-speed broadband links (fast downstream and slow upstream), which would limit the available bandwidth from nearby peers.

How many peers are needed for good performance? Achieving good peer efficiency in a peer-assisted download requires a sufficient number of peers who have a copy of the requested file. We now ask how many copies are needed to achieve a good peer efficiency.

NetSession does not use predictive caching—i.e., a peer only downloads a file when it is requested by the local user. Once a file has been downloaded, the peer keeps it in a local cache for a certain amount of time and informs the control plane that it is willing to upload this file to other peers (if uploading is enabled). This creates entries in the DN log, and we counted these entries for each file to estimate how many copies of that file were available. We also calculated the average peer efficiency for each file.

Figure 5 shows the average peer efficiency as a function of the number of copies registered during the trace (the error bars show the 20th and 80th percentile). We found that, with less than 50 registered copies, peer efficiency was below 10%, but rose rapidly after that, and reached 80% for approximately 10,000 copies.

Another way to look at this question is to ask how many *peers* need to assist in a given download to achieve the highest efficiency. This question is difficult to answer precisely because peer efficiency depends on a variety of other factors, including the size of the object and the network connections of the peers, so we can only hope for a rough trend (with large error bars). To look for this trend, we grouped the downloads by the number of peers that the DN initially suggested to the downloading peer, and we computed the average peer efficiency for each group. Figure 6 shows our results. We find that 80% peer efficiency is generally reached with about 25–30 peers. This is consistent with earlier results for BitTorrent [5, 14], which also requires a few tens of peers to achieve good performance.

Are peer-assisted downloads less reliable? To answer this question, we examined the eventual outcome of the downloads initiated in our trace. The logs can show three kinds of outcomes: a download can complete, it can fail, or it can be aborted/paused by the user and never resumed. When a download fails, the log also shows a specific cause; we divided these into system-related causes (e.g., too many corrupted content blocks) and other causes (e.g., the user's disk is full).

We find that 94% of the infrastructure-only downloads eventually complete, but only 92% of the peer-assisted downloads. At first glance, this discrepancy suggests that peer-assisted downloads are slightly less reliable. However, the rate of system-related failures is very small (0.1% for infrastructure-only versus 0.2% for peer-assisted), but infrastructure-only downloads are paused or terminated less often than peer-assisted ones (3% versus 8%).

A closer look reveals that the termination rate increases with the length of the download. While the download *speed* for infrastructure-only and peer-assisted downloads is approximately the same, the latter are typically used for larger files (see Figure 3(a) earlier), which naturally take longer to download. As Figure 7 shows, larger downloads are terminated more often—hence the discrepancy.

5.3 Do peers improve global coverage?

In theory, a hybrid CDN has another advantage over its infrastructure-based counterparts: it may be easier to obtain a globally distributed population of peers than to establish a truly global infrastructure. Thus, it should be possible for a hybrid system to provide better service for customers in under-served regions, where the closest infrastructure nodes are far away, but peers may be nearby.

To test whether this is the case for NetSession, we aggregated the completed downloads on a per-country basis, and we counted the number of bytes served by the infrastructure and by the peers. We then classified each country into one of three groups, which are shown as colored dots in Figure 8: countries where the infrastructure serves bytes more than the peers (circle), between 50% and 100% of the peers (plus), or less than 50% of the peers (square). Since not all content providers use peer-assisted downloads, and the popularity of the providers varies between regions, we show results for one typical, p2p-enabled provider here.

We find that, for NetSession, the picture is mixed: although the peers tend to contribute more in some regions, such as Africa and South America, the contributions do not vary much overall. We suspect that this is because NetSession relies on edge servers from Akamai's main CDN, which already has very good coverage around the globe.

Another potential benefit of a large peer population is that downloading peers might find a copy of the requested content within their local network, e.g., in a corporate LAN. In October 2012 this case appears to have been rare, but this could change, e.g., when NetSession is used to distribute large software updates.

5.4 Summary

NetSession demonstrates that peer-assisted CDNs can indeed deliver the key benefits of both peer-to-peer and infrastructure-based CDNs: they can offload a considerable fraction of the traffic to the peers, without a significant loss of speed or reliability. NetSession's 80% peer efficiency depends on many factors that may be different in other systems, but it provides a lower bound on what peer-assisted CDNs can achieve in a large-scale commercial deployment.

6. RISKS

We now turn to the question of how well NetSession, as an instance of a peer-assisted CDN, can avoid the risks inherent in this architecture. We again focus on two aspects we can examine quantitatively using our trace: 1) whether NetSession affects the traffic balance of ISPs, and 2) whether the presence of user-managed machines is causing problems.

6.1 Do ISPs suffer from NetSession?

The impact of peer-assisted CDNs on ISPs, and the potential conflict of interest between the two, has been a cause of some concern [17, 24, 35]. To determine how this conflict affects NetSession, we used detailed traffic statistics in our trace, which included, for each peer-assisted download, the GUID of each peer that has sent any content bytes to the downloading client. We first used the login data to map each GUID to the IP address it was using at the time, and then we used the EdgeScape data to map the IP address to the number of the Autonomous System (AS) to which it belongs. The result is a set of (N, AS_1, AS_2) tuples, which describe a flow of N bytes from AS_1 to AS_2. We aggregated the flows per AS, and we report our results at the granularity of ASes. We include only the content bytes, but not the packet headers or the protocol overhead; including the latter would add a small constant factor. We also neglect bytes sent by the infrastructure, since these would be sent by an infrastructure-based CDN as well, and since most of them would normally be sent from an edge server within the same AS anyway.

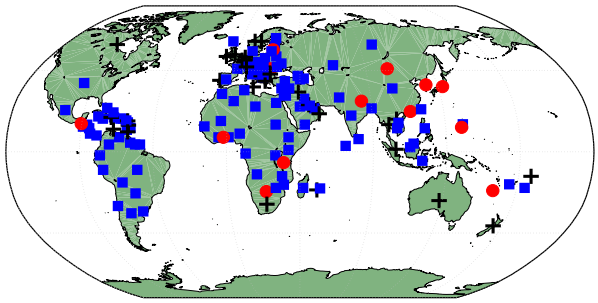


Figure 8: Peer contributions in different regions (for one exemplary content provider).

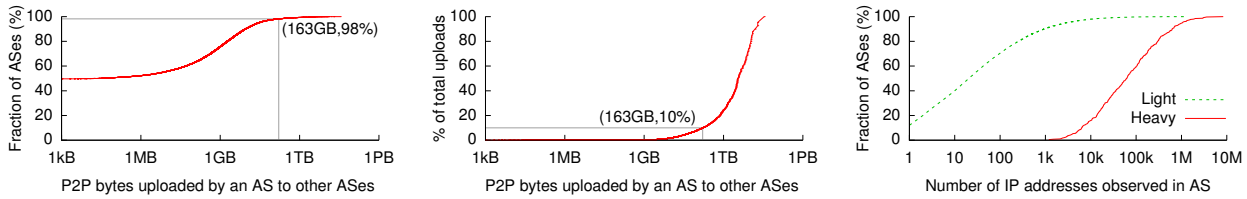
Which ASes are sending the most? Overall, our trace shows that 895 TB of content bytes were sent peer-to-peer, and that the peers involved in these transmissions were located in roughly 22,000 different ASes. (The number is different from that in Table 1 because not all GUIDs participated in a p2p transaction.) 18% of this traffic was sent between peers in the same AS; we do not further consider such intra-AS traffic here because it does not affect the inter-AS bandwidth cost.

To determine how the inter-AS traffic was distributed across the ASes, we sorted the ASes by the number of bytes they sent to peers in other ASes. Figure 9(a) shows the result as a CDF. Roughly half of the ASes did not send any inter-AS bytes at all, and 98% sent less than 163 GB over the entire month. However, there is a "heavy tail" of ASes that sent substantially more; the top contributor sent approximately 34.2 TB.

Figure 9(b) shows a different perspective on this result: a point (x, y) on this plot indicates that the cumulative contribution of ASes contributing less than x bytes amounted to y% of the total inter-AS p2p traffic. This plot shows that 98% of the ASes contributed just 10% of the bytes; the remaining 2% (394 ASes) were responsible for 90% of the p2p traffic. We refer to the former as "light" and "heavy" uploaders, respectively. Figure 9(c) shows that this uneven distribution has a natural explanation; the heavy uploaders simply contain a lot more peers.

How balanced is the traffic? Sending a large number of bytes to its neighbors is not necessarily expensive for an AS, as long as it receives an equally large number of bytes from them; balanced connections with peering ASes may be considered "settlement free". To examine whether NetSession's p2p traffic affects this balance, we separately counted the inter-AS bytes each AS sent and received. Figure 10 shows the result as a scatterplot. Note that the scales are logarithmic; since zero does not appear on a logarithmic scale, the points for ASes that have not uploaded and/or downloaded anything appear near the axes.

Figure 10 shows that ASes often have a substantial *relative* imbalance; many receive several Gigabytes but send



(a) Inter-AS traffic (b) Contributions of different ASes (c) Distinct IP addresses per AS

Figure 9: Traffic balance results.

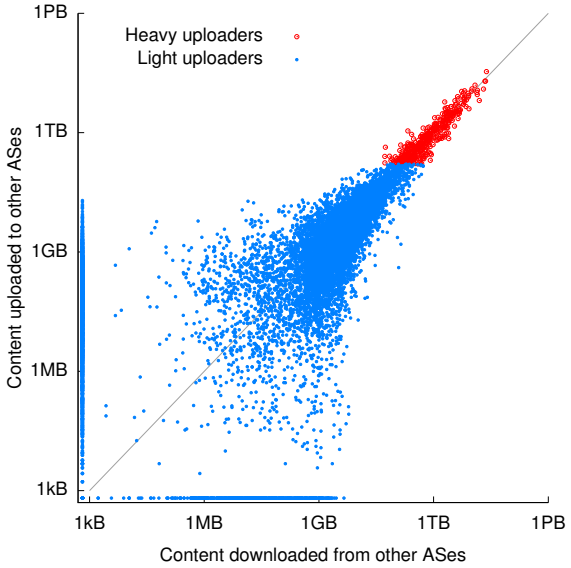


Figure 10: p2p bytes uploaded vs. downloaded for an AS. Only the bytes which cross AS boundaries are considered here.

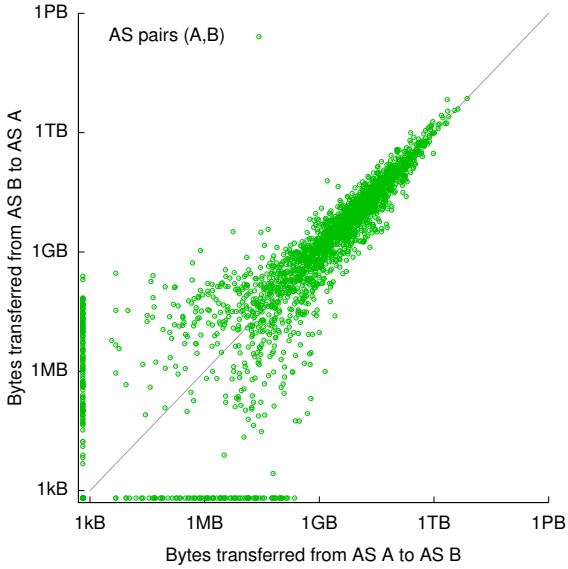


Figure 11: Traffic balance on AS-to-AS links.

hardly anything, or vice versa. However, this imbalance mostly affects ASes that are not carrying very much NetSession p2p traffic anyway. The traffic of the "heavy" uploaders (which appear in red towards the upper right end of the plot) tends to be well-balanced, i.e., they usually receive as much as they send. At a high level, this is not unexpected: if uploaders and downloaders are distributed roughly uniformly, the resulting traffic is naturally balanced. However, there are many effects that *could* bias this pattern—for instance, clients in certain ASes might have a higher upstream bandwidth and might therefore be chosen to upload more often. NetSession avoids such biases in part by limiting the number of times a peer will upload a file it has locally cached.

Note that we have only considered the origin and the destination of the traffic here. We cannot reliably estimate the effect of transit traffic because our data set does not contain traceroutes. Based on CAIDA's AS topology data [6], we estimate that out of the total inter-AS p2p bytes exchanged between the "heavy" uploaders, approximately 35% were exchanged between those that had a direct connection with each other.

So far, we have seen that the overall incoming traffic of an AS is largely balanced by its overall outgoing traffic. However, a *pairwise* imbalance could still exist between pairs of ASes. Figure 11 shows that this is not the case: analogous to above, the figure contains the distribution of p2p bytes exchanged between pairs of "heavy" uploaders that had a direct connection (though not necessarily a peering relationship) with each other. The trend is similar to Figure 10: heavy contributors tend to have a traffic balance that is roughly even.

We emphasize that our results should *not* be interpreted as saying that NetSession's traffic has no impact on ISPs. Our data only describes the traffic, and not the cost that ISPs may or may not charge each other for carrying that traffic. This cost depends on a variety of other factors as well, including the business relationships between the ISPs. Nevertheless, it is likely that a large traffic imbalance would eventually affect the business relationship and/or the cost, and our results do show that this particular problem does not seem to be occurring.

6.2 Impact of user-managed machines

How strong is the mobility-related churn? Unlike the servers in an infrastructure-based CDN, the NetSession peers can move; for instance, users can install NetSession on a laptop that they take to work in the morning and back home in the evening. Overall, we observed connections from 133.7 million different IP addresses in 31,190 different autonomous systems (ASes); 80.6% of the GUIDs connected from a single AS, 13.4% from two different ASes, and 6% from more than two ASes. The DNs must update their directories to keep track of this mobility, but the corresponding workload is moderate: on average, the control plane receives 20,922 new connections per minute.

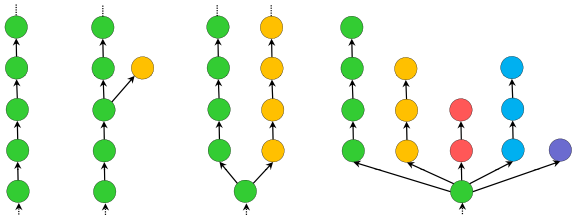


Figure 12: Expected sequence of secondary GUIDs (left) and common nonlinear patterns.

As an aside, to get a rough sense of the corresponding physical mobility, we computed for each GUID the two geolocations that were farthest apart. We found that 77% remained within 10 km, and that 23% were more than 10 km apart. We note that the actual mobility may be somewhat lower; for instance, if a user occasionally opens a VPN connection to her corporate headquarters, the geolocation of the latter may be far away.

Can malicious peers do damage? Unlike a centrally managed infrastructure, which mostly needs to be protected against external attackers, a peer-assisted CDN also needs protection against attacks from *within*—each user physically controls his or her machine and can change the software on it at will. Previous work has shown that this can lead to new vulnerabilities that are unique to hybrid architectures, such as accounting attacks [1], in which compromised peers misreport the amount of service they have provided, in an attempt to distort the reports the CDN provides to its customers. As mentioned in Section 3.5, NetSession relies on data from the (trusted) edge servers to detect such attacks and to filter out incorrect reports. Measures are in place to make modification of the client-side executable difficult, but we have seen users experiment with manually modifying data in configuration files.

GUID cloning and re-imaging: Even in the absence of attacks, the designer of a hybrid CDN must consider more failure modes than in an infrastructure-based system. Here, we cannot provide an exhaustive discussion due to lack of space, but we discuss an illustrative example. While analyzing an earlier set of traces, we noticed that certain GUIDs showed unusual behavior: they were logging in very frequently, sometimes from IP addresses in different countries and even on different continents. We initially suspected international travel, but some users appeared to be traveling at implausible speeds. We eventually suspected that some installations had to be sharing the same GUID.

To investigate further, we modified the NetSession software to support a random 160-bit "secondary GUID", which is chosen freshly every time the software *starts* (unlike the primary GUID, which is chosen when the software is first installed), and to report the last five secondary GUIDs to the control plane upon login. A normal NetSession installation would report overlapping sequences of secondary GUIDs, such as `5 4 3 2 1`, `6 5 4 3 2`, and so on. We then collected and analyzed the secondary GUIDs on a sample of 8 control plane servers, grouped them by primary GUID, and constructed graphs in which vertices represent secondary GUIDs and edges connect GUIDs that follow each other in a login entry. In total, we obtained 17.7 million connected graphs with at least three vertices.

99.4% of the graphs were linear chains (as in our example: $1 \to 2 \to 3 \to \ldots$), which would be expected for a normal NetSession installation. But the remaining 0.6% were trees, indicating that the corresponding installation had at some point been rolled back to an earlier state! Figure 12 shows the most common nonlinear patterns: one long branch with a single, one-vertex short branch (46.2%), two long branches (6.2%), and several short or medium branches (23.5%). We suspect that the first pattern corresponds to a failed software update and the second pattern to a restored backup; the third pattern might be due to re-imaging, e.g., in an Internet café that restores its machines every night, or due to cloning, e.g., in a large IT department that initializes new workstations from a master image. We also observed several highly irregular patterns for which we currently have no explanation.

Recall that this example was merely meant to illustrate that hybrid systems must deal with user actions that would not normally occur in a centrally managed infrastructure. However, this specific example was of some practical importance, since the number of distinct GUIDs plays a role in NetSession's internal accounting.

7. RELATED WORK

CDN measurement studies: There are several prior studies of content-distribution systems, including both peer-to-peer [13,26] and infrastructure [32] systems. The systems studied include p2p streaming systems [15, 28, 34] and one commercial hybrid CDN, LiveSky [36]. LiveSky delivers live-streaming video, so its architecture is necessarily different from NetSession's. To the best of our knowledge, the present paper is the first measurement study of a commercial hybrid CDN for static content.

Hybrid CDNs: The idea of combining a CDN infrastructure with peer-to-peer elements has been explored both in research [12, 22] and in industry [33, 34, 36]; see [20] for a survey. The Antfarm system [22], in particular, has some similarities to NetSession. Antfarm combines peer-to-peer swarms with a coordinator, which carefully directs bandwidth provided by the infrastructure servers to maximize the aggregate bandwidth of the swarms. NetSession's control plane plays a similar role but, unlike Antfarm's coordinator, it does not implement an explicit incentive mechanism. Work on the potential benefits, specifically [16, 17], has already been discussed in Section 5; likewise, work on the inherent risks, such as [1], has been discussed in Section 6.

Impact on ISPs: Several papers [7, 17, 24, 35] have noted a potential conflict of interest between hybrid CDNs and ISPs because the former potentially increase the upstream traffic, and thus the cost, of the latter. The work in [7] demonstrated that a peer selection algorithm using CDN-based DNS "hints" can significantly reduce cross-ISP traffic. However, the general conclusion in this work is that the CDN can avoid a large impact on ISPs by using a simple locality-aware peer selection strategy.

Incentives: Swarming protocols like BitTorrent [8] typically include explicit incentives because they assume that users will not contribute bandwidth unless they can benefit from it. There is a large body of work on understanding specific incentive systems, e.g., [18,23], and on building systems that can deliver them more robustly, e.g., [19]; Dandelion [30] even introduces an infrastructure component similar to NetSession's control plane, although its main purpose is to implement a fair-exchange mechanism. The existence and efficient operation of NetSession does not contradict the

basic assumptions on which these systems are based, but it does suggest that the users' actual motivations are more complex, and that avoiding bandwidth contributions may not be among their primary concerns.

8. CONCLUSION

In this paper, we have examined the risks and benefits of "hybrid" CDN architectures, which rely on a managed infrastructure but also include a peer-to-peer element to leverage resources contributed by clients. To determine how well a practical system can deliver the benefits (and avoid the risks), we have presented a measurement study of NetSession, a large commercial CDN that uses this architecture.

Our results show that NetSession is able to deliver the key benefits of a hybrid architecture: it can offload a high fraction (70–80%) of the traffic to peers, but it can *also* offer good performance and high reliability. NetSession appears to avoid a key risk—tilting the traffic balance of ISPs—but it does face other challenges that are inherent in its hybrid architecture, e.g., with respect to security or manageability.

Overall, our findings suggest that a hybrid architecture is an attractive design point for a CDN. The infrastructure and the peers can deliver many of their key benefits, and they can complement one another to avoid many of their key weaknesses. NetSession's performance shows what is possible in this space.

Acknowledgments

We thank our shepherd Craig Partridge and the anonymous reviewers for their comments and suggestions. This work was supported in part by NSF grants CNS-1040672, CNS-1054229, and CNS-1065130, by AFRL grants FA8750-11-1-0262 and FA8750-10-2-0193, and by the Max Planck Society.

9. REFERENCES

[1] P. Aditya, M. Zhao, Y. Lin, A. Haeberlen, P. Druschel, B. Maggs, and B. Wishon. Reliable client accounting for hybrid content-distribution networks. In *Proc. NSDI*, Apr. 2012.

[2] Akamai. EdgeScape brochure. `http://www.akamai.com/dl/brochures/edgescape.pdf`.

[3] Akamai Technologies. NetSession design principles. `http://www.akamai.com/html/solutions/client_design_principles.html`.

[4] R. Bhagwan, S. Savage, and G. M. Voelker. Understanding availability. In *Proc. IPTPS*, 2003.

[5] A. R. Bharambe, C. Herley, and V. N. Padmanabhan. Analyzing and improving a BitTorrent network's performance mechanisms. In *Proc. INFOCOM*, 2006.

[6] CAIDA. Archipelago measurement infrastructure. `http://www.caida.org/projects/ark/`.

[7] D. R. Choffnes and F. E. Bustamante. Taming the Torrent. In *Proc. SIGCOMM*, 2008.

[8] B. Cohen. Incentives build robustness in BitTorrent. In *Proc. P2PEcon*, June 2003.

[9] S. A. Crosby and D. S. Wallach. An analysis of BitTorrent's two Kademlia-based DHTs. Technical Report TR-07-04, Department of Computer Science, Rice University, May 2007.

[10] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl. Globally distributed content delivery. *IEEE Internet Computing*, 6(5):50–58, Sept. 2002.

[11] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing residential broadband networks. In *Proc. IMC*, Oct. 2007.

[12] M. J. Freedman. Experiences with CoralCDN: A five-year operational view. In *Proc. NSDI*, 2010.

[13] K. P. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan. Measurement, modeling and analysis of a peer-to-peer file-sharing workload. In *Proc. SOSP*, Oct. 2003.

[14] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of BitTorrent-like systems. In *Proc. IMC*, 2005.

[15] X. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross. A measurement study of a large-scale P2P IPTV system. *Multimedia, IEEE Trans.*, 9(8):1672–1687, Dec. 2007.

[16] C. Huang, A. Wang, J. Li, and K. W. Ross. Understanding hybrid CDN-P2P: Why Limelight needs its own Red Swoosh. In *NOSSDAV*, 2008.

[17] T. Karagiannis, P. Rodriguez, and K. Papagiannaki. Should Internet service providers fear peer-assisted content distribution? In *Proc. IMC*, 2005.

[18] D. Levin, K. LaCurts, N. Spring, and B. Bhattacharjee. BitTorrent is an auction: Analyzing and improving BitTorrent's incentives. In *Proc. SIGCOMM*, 2008.

[19] H. C. Li, A. Clement, M. Marchetti, M. Kapritsos, L. Robison, L. Alvisi, and M. Dahlin. FlightPath: Obedience vs. choice in cooperative services. In *Proc. OSDI*, 2008.

[20] Z. Lu, Y. Wang, and Y. R. Yang. An analysis and comparison of CDN-P2P-hybrid content delivery system and model. *JCM*, 7(3):232–245, 2012.

[21] W. E. Mackay. Triggers and barriers to customizing software. In *Proc. CHI*, Apr. 1991.

[22] R. S. Peterson and E. G. Sirer. Antfarm: efficient content distribution with managed swarms. In *Proc. NSDI*, 2009.

[23] M. Piatek, T. Isdal, T. Anderson, and A. Krishnamurthy. Do incentives build robustness in BitTorrent? In *Proc. NSDI*, 2007.

[24] M. Piatek, H. V. Madhyastha, J. P. John, A. Krishnamurthy, and T. Anderson. Pitfalls for ISP-friendly P2P design. In *Proc. HotNets*, 2009.

[25] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing. Session Traversal Utilities for NAT (STUN). RFC 5389, Oct. 2008.

[26] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy. An analysis of Internet content delivery systems. In *Proc. OSDI*, 2002.

[27] S. Saroiu, K. P. Gummadi, and S. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proc. MMCN*, Jan. 2002.

[28] T. Silverston and O. Fourmaux. Measuring P2P IPTV systems. In *Proc. NOSSDAV*, 2007.

[29] M. Sirivianos, J. H. Park, R. Chen, and X. Yang. Free-riding in BitTorrent networks with the Large View exploit. In *Proc. IPTPS*, Feb. 2007.

[30] M. Sirivianos, J. H. Park, X. Yang, and S. Jarecki. Dandelion: Cooperative content distribution with robust incentives. In *Proc. USENIX ATC*, 2007.

[31] J. Spool. Do users change their settings? `http://www.uie.com/brainsparks/2011/09/14/do-users-change-their-settings/`.

[32] K. Sripanidkulchai, B. Maggs, and H. Zhang. An analysis of live streaming workloads on the Internet. In *Proc. IMC*, 2004.

[33] Velocix. Network architecture. `http://www.velocix.com/network_architecture.php`.

[34] L. Vu, I. Gupta, K. Nahrstedt, and J. Liang. Understanding overlay characteristics of a large-scale peer-to-peer IPTV system. *ACM Tr. Multim. Comp. Comm. Appl.*, 6:31:1–31:24, 2010.

[35] H. Xie, R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Provider portal for applications. In *Proc. SIGCOMM*, 2008.

[36] H. Yin, X. Liu, T. Zhan, V. Sekar, F. Qiu, C. Lin, H. Zhang, and B. Li. Design and deployment of a hybrid CDN-P2P system for live video streaming: experiences with LiveSky. In *Proc. MM*, 2009.