System Designs for Bulk and User-generated Content Delivery in the Internet

Massimiliano Marcon

Dissertation

zur Erlangung des Grades
des Doktors der Ingenieurwissenschaften
der Naturwissenschaftlich-Technischen Fakultäten
der Universität des Saarlandes

Eingereicht Saarbrücken, 13. Juli 2011

Dekan:	Prof. Dr.-Ing. Holger Hermanns
Vorsitzender des Promotionsausschusses:	Prof. Dr. Michael Backes
Erstgutachter:	Prof. Dr. Peter Druschel
Zweitgutachter:	Krishna P. Gummadi, PhD
Tag des Kolloquiums:	4. November 2011

Abstract

This thesis proposes and evaluates new system designs to support two emerging Internet workloads: (a) bulk content, such as downloads of large media and scientific libraries, and (b) user-generated content (UGC), such as photos and videos that users share online, typically on online social networks (OSNs).

Bulk content accounts for a large and growing fraction of today's Internet traffic. Due to the high cost of bandwidth, delivering bulk content in the Internet is expensive. To reduce the cost of bulk transfers, I proposed traffic shaping and scheduling designs that exploit the delay-tolerant nature of bulk transfers to allow ISPs to deliver bulk content opportunistically. I evaluated my proposals through software prototypes and simulations driven by real-world traces from commercial and academic ISPs and found that they result in considerable reductions in transit costs or increased link utilization.

The amount of user-generated content (UGC) that people share online has been rapidly growing in the past few years. Most users share UGC using online social networking websites (OSNs), which can impose arbitrary terms of use, privacy policies, and limitations on the content shared on their websites. To solve this problem, I evaluated the feasibility of a system that allows users to share UGC directly from the home, thus enabling them to regain control of the content that they share online. Using data from popular OSN websites and a testbed deployed in 10 households, I showed that current trends bode well for the delivery of personal UGC from users' homes. I also designed and deployed Stratus, a prototype system that uses home gateways to share UGC directly from the home.

Kurzdarstellung

Schwerpunkt dieser Doktorarbeit ist der Entwurf und die Auswertung neuer Systeme zur Unterstützung von zwei entstehenden Internet-Workloads: (a) Bulk-Content, wie zum Beispiel die Übertragung von großen Mediendateien und wissenschaftlichen Datenbanken, und (b) nutzergenerierten Inhalten, wie zum Beispiel Fotos und Videos, die Benutzer üblicherweise in sozialen Netzwerken veröffentlichen.

Bulk-Content macht einen großen und weiter zunehmenden Anteil im heutigen Internetverkehr aus. Wegen der hohen Bandbreitenkosten ist die Übertragung von Bulk-Content im Internet jedoch teuer. Um diese Kosten zu senken habe ich neue Scheduling- und Traffic-Shaping-Lösungen entwickelt, die die Verzögerungsresistenz des Bulk-Verkehrs ausnutzen und es ISPs ermöglichen, Bulk-Content opportunistisch zu übermitteln. Durch Software-Prototypen und Simulationen mit Daten aus dem gewerblichen und akademischen Internet habe ich meine Lösungen ausgewertet und herausgefunden, dass sich die Übertragungskosten dadurch erheblich senken lassen und die Ausnutzung der Netze verbessern lässt.

Der Anteil an nutzergenerierten Inhalten (user-generated content, UGC), die im Internet veröffentlicht wird, hat in den letzen Jahren ebenfalls schnell zugenommen. Meistens wird UGC in sozialen Netzwerken (online social networks, OSN) veröffentlicht. Dadurch sind Benutzer den willkürlichen Nutzungsbedingungen, Datenschutzrichtlinien, und Einschränkungen des OSN-Providers unterworfen. Um dieses Problem zu lösen, habe ich die Machbarkeit eines Systems ausgewertet, anhand dessen Benutzer UGC direkt von zu Hause veröffentlichen und die Kontrolle über ihren UGC zurückgewinnen können. Meine Auswertung durch Daten aus zwei populären OSN-Websites und einem Feldversuch in 10 Haushalten deutet darauf hin, dass angesichts der Fortschritte in der Bandbreite der Zugangsnetze die Veröffentlichung von persönlichem UGC von zu Hause in der nahen Zukunft möglich sein könnte. Schließlich habe ich Stratus entworfen und entwickelt, ein System, das auf Home-Gateways basiert und mit dem Benutzer UGC direkt von zu Hause veröffentlichen können.

Contents

List of Figures

List of Tables

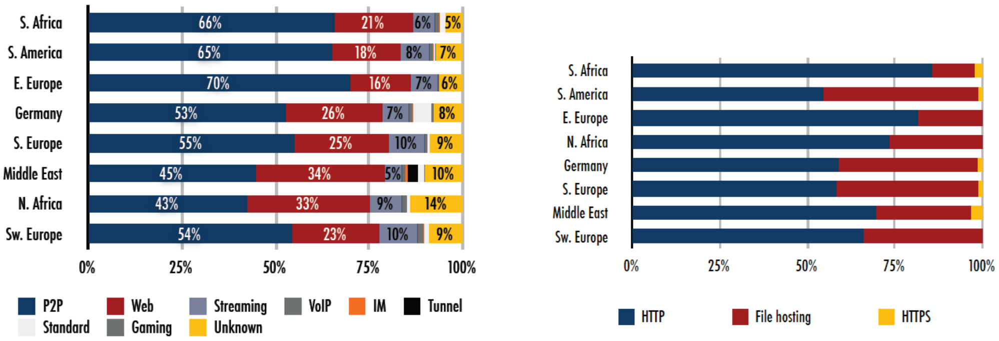
1. Introduction

Since its inception in the late 1960s, the Internet has been used to exchange information between users. In fact, shortly after its creation, the initial ARPANET network was already carrying email messages between its users, and email quickly became one of the first major applications. As the Internet kept on growing through the 1980s and early 1990s, it began to be used not only for message exchange and research, but also for content delivery. The need for easy and rapid access to content sparked further technological advances, the most important being the development of the World Wide Web in the late 1980s. At that time, the Internet was still government funded and thus restricted to educational and research purposes. Later on, in the early 1990s, the Internet opened to commercial use and has continued to steadily grow in size. Today, the Internet is a worldwide network that comprises hundreds of millions of nodes and is used by almost two billion people worldwide.

While initially used to distribute small text documents of a few kilobytes each, the increase in available network bandwidth has by now made the Internet an attractive delivery channel for much larger content. Today, the Internet is routinely being used to deliver large content like DVD movies, large scientific datasets, and entire disk backups. We call this emerging content, which can reach sizes of many Gigabytes or even Terabytes, *bulk content*. At the same time, and orthogonally to bulk content, emerging Web 2.0 applications like *online social networking (OSN)*, combined with the widespread availability of digital devices like photo and video cameras, have made it easy for all users to upload and distribute their personal, *user-generated content (UGC)* through the Internet. User-generated content is produced by many independent and non-professional sources. Moreover, some of this UGC is of a personal nature and therefore of interest to a relatively small number of people. User-generated content contrasts with the traditional content distributed online, which typically originates from a few professional sources, like artists, newspapers, or universities, and is intended for a large number of users. This thesis deals with the consequences for the design of content distribution systems of these two emerging types of content: bulk and user-generated content. We now outline the salient characteristics of these two types of content.

1.1. Emerging Internet workloads: challenges and opportunities

The motivation for the work behind this thesis is the recent shift in Internet workloads towards two emerging classes of content: bulk and user-generated. These new classes of content differ significantly from the traditional web-based workloads that have dominated the Internet thus far.



(a) Internet traffic by protocol

(b) HTTP breakdown

Figure 1.1: Bulk traffic represents a large and growing fraction of total Internet traffic: A large fraction of Internet traffic is generated by P2P applications, which generate bulk traffic (a). Moreover, a large fraction of web traffic is ascribed to file hosting application (b), which are also bulk in nature (source: Ipoque Internet Study 2008/2009 [ipo]).

1.1.1. Bulk content workloads

Distribution of movies and music [Elm08], sharing of large scientific repositories [Uni07], backups of personal data [AMA], and P2P file sharing [BIT] are all examples of Internet bulk applications. Bulk transfers account for a large and growing fraction of bytes transferred across the Internet. This is backed by recent studies of Internet traffic in commercial and research backbones [KBB$^+$04] as well as academic [Bkc02] and residential [CFEK06, MFPA09] access networks. Figure 1.1(a) shows the composition of Internet traffic as observed by Ipoque, a leading vendor of deep packet inspection (DPI) devices [ipo], in 2008/2009. The data reveals that in most areas of the world P2P bulk applications already account for a majority of Internet traffic. The second largest contributor to Internet traffic is Web, for which Figure 1.1(b) shows a classification into application types. The result is that a considerable and growing fraction of web traffic is attributed to emerging file hosting services, which also generate bulk traffic. These observations taken together suggest that bulk applications often account for a majority of the traffic exchanged over the Internet, and this fraction is growing because file hosting and file sharing services are becoming more popular [AMD09].

Despite rising demand for bulk content, the cost of bandwidth in the Internet remains high and consequently bulk data transfers remain expensive. For example, the monthly cost of raw transit bandwidth, depending on the region of the world, ranges from $30K to $90K per Gbps [LSRS09]. Data transfers to and from Amazon's S3 online storage service cost 10 to 17 cents per GB without any bandwidth guarantees [AMA]. These high transit costs are forcing many edge ISPs to discourage the use of bulk data applications by rate-limiting or blocking bulk flows [DMHG08].

Although bulk traffic is expensive when we only consider the raw amount of bandwidth it consumes, it also has less strict requirements on the delivery of network packets when compared to interactive applications like voice over IP (VoIP) or web browsing. While most interactive applications require that packets be delivered with low, predictable latency to ensure satisfactory user experience, the main performance metric for most bulk transfers is raw average throughput. In other words, as long as bulk transfers complete on time, the delay incurred by single network packets is irrelevant. Since packets from bulk transfers can tolerate delays, it is possible to make them wait in order to speed up delivery of interactive traffic. This presents an interesting opportunity for lowering the cost of bulk transfers by optimizing resource consumption, giving precedence to interactive traffic over bulk, which will then use network resources only when more demanding traffic doesn't need them.

1.1.2. User-generated workloads (UGC)

The content on the Internet, although consumed by a large number of end users, has traditionally been generated and distributed by only a few professional sources. Examples of such content include news items, as well as professional movies, photographs, and music. The most prevalent way of delivering this kind of content is through the client-server model adopted by most Internet applications, where content is delivered by a centralized server located in the network core to potentially many network clients, located at the edge of the network. A prominent exception to the server-client model is P2P file sharing systems [BIT, emua], which allow content to be directly exchanged between users located at the network edge. Unlike client-server architectures, P2P file sharing systems involve end users in the distribution of content. However, historically, they have still been used to distribute professional content like movies and music, often protected by copyright.

Despite the prevalence of professional content in the Internet, the fraction of content that originates from non-professional sources has been recently increasing. This has been fostered by the increased availability of video and photo cameras, as well as emerging Web 2.0 applications like online social networking (OSN), which have made it easy for all Internet users to upload and share content with other users, regardless of their location. This new kind of content is usually referred to as user-generated content (UGC). Popular examples of user-generated content that is shared on the Internet today include text messages [twi], photos [Fac09], and videos [Youa]. This workload is inherently distributed, because it is generated *and* consumed at the network edge. In many cases, UGC is personal in nature, and its intended recipients are the uploader's acquaintances or family. This contrasts with traditional content generated by professional sources (e.g. movie and TV studios, newspapers, musicians), which typically strive to reach the largest possible audience. In most cases, users upload their UGC to centralized social networking sites in order to share it with other users. For example, Facebook, a very popular OSN website, stores more than 15 billion user photos, making it the biggest photo-sharing site on the Web [Fac09].

OSN sites are convenient for users, because they host uploaded content on well-provisioned servers and make it readily available for other users to view. However, when uploading their content to centralized websites, users are partially giving away control of the content they are sharing. Drawbacks of centralized systems include: constraints on what can be shared, copyrights losses, and volatility of privacy policies from OSN providers. Moreover, OSN providers find it increasingly challenging to store and deliver a growing amount of low-popularity OSN content using the traditional web-delivery infrastructure, such as web caches and CDNs, which is not optimized for delivering unpopular content.

Although a large amount of UGC is currently shared on OSN sites, the way in which UGC is accessed lends itself to alternative forms of sharing. In particular, since most of a user's UGC is probably of a personal nature and of interest to a relatively small audience, such as family and friends, the content can be directly distributed from the user's home if available resources (such as network and storage) allow it. This idea is gaining plausibility thanks to current trends such as the increasing capacity of residential access network, lower storage costs, and the presence of always-on home gateways in users' homes that can act as servers. Delivering UGC from home has the potential to address the privacy and control concerns associated with storing personal data on centralized OSNs.

1.2. Contributions

The contributions of this thesis are (a) proposing and estimating the performance of new designs for traffic management of bulk traffic and (b) proposing and estimating the performance of new systems for sharing user-generated content.

1.2.1. Network-level designs for traffic management of bulk traffic

We propose and evaluate two new designs for the management of bulk traffic. These designs are intended for deployment by ISPs. The first design is intended for access ISPs, which buy transit and have an incentive to traffic shape bulk traffic from their customers with the goal of reducing traffic costs and the risk of congestion. The second design is to be implemented by transit ISPs in the network core, which have an incentive to maximize the utilization of their costly network infrastructure to create additional revenue streams.

Traffic shaping of bulk transfers in access ISPs We study different traffic shaping techniques that access ISPs can use to reduce their peak utilization, thus lowering bandwidth costs as well as the risk of network congestion. All the techniques we study shape bulk flows, because they consume most of the bandwidth and can be delayed as long as their completion time remains sufficiently short. After evaluating various existing traffic shaping techniques, we propose an ideal new technique that achieves good peak reduction and minimizes the local impact on targeted flows. However, we also find that this technique results in large global performance losses for bulk flows that traverse multiple traffic shapers deployed by different ISPs. To solve this problem, we propose staging the transfers. With staging, end-to-end bulk transfers are broken into multiple subtransfers, each traversing only one traffic shaper. Between subtransfers, network traffic has to be temporarily stored in the network. Because each subtransfer composing a staged transfer only traverses at most one traffic shaper, staging of bulk transfers eliminates the performance loss caused by multiple traffic shapers on the same end-to-end path.

Opportunistic bulk content delivery in transit ISPs We describe the design and evaluate the potential of NetEx, an opportunistic bulk delivery system that transit ISPs can deploy in their backbones. NetEx delivers bulk data by opportunistically tapping capacity that is not used by current traffic. Since bulk traffic can be delayed as long as its completion time remains low, it can wait for bandwidth to become available and thus use network resources opportunistically. NetEx is able to allocate a substantial portion of spare network capacity to bulk applications, and at the same time leave current traffic completely unaffected. With NetEx, transit ISPs can tap previously unused bandwidth and potentially market it by offering a traffic class that does not require strict SLAs. We quantify the potential performance of NetEx through detailed simulations driven by data from real commercial and research backbones.

1.2.2. Home-based distribution of UGC

We make the following contributions towards new content distribution systems that distribute user-generated content (UGC) from home.

Feasibility study of UGC distribution from the home environment We take a first step towards answering the question of whether UGC can be distributed directly from user homes. Using traces from popular OSN websites and from wireless routers deployed in a number of households, we characterize both UGC workloads and the resources available in the home environment. We then evaluate the extent to which existing UGC workloads can be delivered from the home environment, with respect to both content availability and performance. We also evaluate the potential of a hybrid system that stores only popular user-generated content on a centralized website, while delivering the remaining, less popular objects from home.

A prototype system for UGC distribution from home As a second step towards UGC distribution from home, we describe the design and implementation of Stratus, a prototype system that enables users to share their personal content with friends and acquaintances directly from home. Our implementation of Stratus runs on currently available wireless routers that users can employ as their Internet home gateways. These routers are equipped with a USB port, and can therefore be connected to commodity mass storage devices. Stratus makes some content stored on these devices available to other users whom the content's owner has granted access permissions to. Users can access Stratus from any browser through a Facebook application and transparently authenticate to the Stratus personal server with their Facebook identities.

1.3. Structure of this thesis

The rest of this thesis is structured as follows. Part I deals with new system designs for bulk traffic management. In particular, Chapter 3 focuses on access ISPs, and presents an analysis of existing as well as new traffic shaping techniques that target bulk flows. We quantify and discuss both the local and global effects of these techniques on Internet traffic. Chapter 4 describes NetEx, a new opportunistic bulk delivery system that transit ISPs can deploy to increase the usage of their backbone networks. Part II discusses new designs for the delivery of emerging user-generated content workloads. In Chapter 5 we present a study of the feasibility of a generic system that delivers UGC from home and outline the design of Stratus, a concrete system that allows users to share their UGC directly from their homes. Finally, Chapter 6 concludes the thesis and outlines future work.

Part I.

Network-level designs for bulk traffic management

2. Bulk data delivery in the Internet

Bulk traffic, such as distribution of media files, scientific data or online backups, account for a large and growing fraction of bytes transferred across the Internet. This trend is backed by recent studies of Internet traffic in commercial and research backbones [KBB$^+$04] as well as academic [Bkc02] and residential [CFEK06, MFPA09] access networks. The prevalence and growth of bulk transfers in the Internet is also confirmed by industry reports on the composition of Internet traffic [ipo]. Finally, the recent emergence of one-click hosting services [AMD09] used to distribute large files may increase the amount of bulk traffic in the near future.

In the first part of this thesis, we propose new techniques that ISPs can use to manage bulk traffic. We first give an overview of the different kinds of ISPs that constitute the Internet, discuss their model of operation and point out their challenges and opportunities with managing bulk traffic.

2.1. Challenges and opportunities for ISPs

ISPs can roughly be subdivided into two classes: access and transit. Figure 2.1 highlights the main differences between these two types of ISPs. Transit ISPs generally don't generate any traffic, but simply connect customer ISPs by carrying Internet traffic through their network backbones. Some transit ISPs, like the regional ISPs shown in Figure 2.1, can in turn buy transit from larger ISPs. Tier-1 ISPs are the only transit ISPs that don't pay for transit. Access ISPs provide Internet access to residential users, corporations, or universities. Access ISPs either generate or terminate Internet traffic because they contain the hosts at the endpoints of most Internet transfers. Typically, access ISPs pay transit ISPs for connectivity to the rest of the Internet.

2.1.1. Access ISPs

Access ISPs provide connectivity to corporate or residential customers, and almost always pay transit ISPs for connectivity. In order to discourage traffic spikes that might congest the network, transit ISPs bill their customers based on near-peak (typically 95^{th}-percentile) utilization [GQX$^+$04]. With 95^{th}-percentile billing, traffic level is periodically sampled, typically every 5-minute. At the end of the billing period, all samples are sorted and the 95^{th}-percentile sample is used to compute the network charges. Figure 2.2 plots the traffic on a university access link and highlights peak and 95^{th} percentile traffic levels. 95^{th}-percentile billing dissuades customers from generating high traffic peaks, because peaks accounting for more than 5% of the billing period will contribute to an increase in network charges.

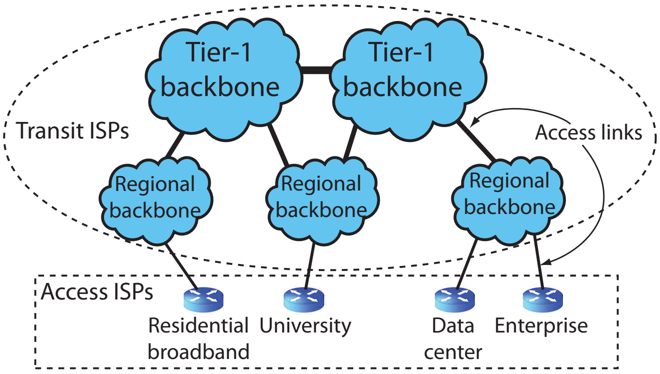


Figure 2.1: High-level structure of the Internet: Access ISPs generate and consume data that is transferred by transit ISPs.

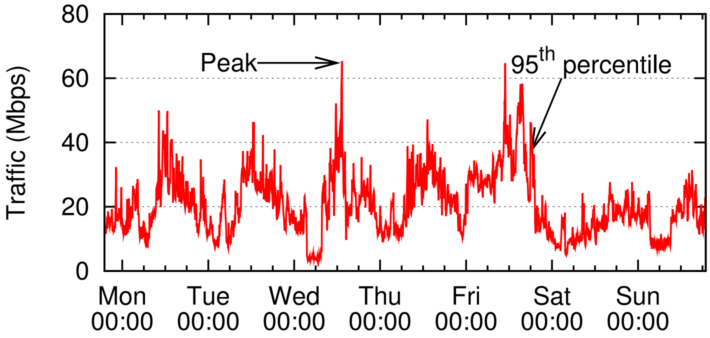


Figure 2.2: Weekly upstream traffic on a University access link: Access ISPs are often charged based on 95^{th} percentile usage.

Because access ISPs are charged based on 95^{th}-percentile utilization, they have incentives to reduce their bandwidth costs through traffic shaping. Access ISPs today deploy a variety of traffic shaping policies, whose main goal is to reduce network congestion and to distribute bandwidth fairly amongst customers [Bel]. This is typically achieved by lowering peak network usage through traffic shaping of either single flows or a subscriber's aggregate traffic. At the same time, ISPs also try to affect as few flows as possible to keep the effect on user traffic low [Röt08].

Because packets from most bulk traffic can tolerate delays that would be unacceptable for interactive applications, it makes sense to consider policies that intelligently delay bulk transfers in order to reduce peak utilization (and thus costs), while at the same time keeping the negative impact on bulk traffic to the necessary minimum.

However, at the time of writing, no known studies investigate the economic benefits of ISPs shaping policies relative to their negative impact on the performance of bulk transfers, and thus their negative impact on customer satisfaction. On the contrary, existing traffic shaping policies are often blunt and arbitrary. For example, some ISPs limit the aggregate bandwidth consumed by bulk flows to a fixed value, independent of the current level of link utilization [Fri02]. A few ISPs even resort to blocking entire applications [DMHG08].

2.1.2. Transit ISPs

Transit ISPs run regional, national, or continental backbones that relay traffic between customer ISPs. Customer ISPs need to make sure that the quality of service offered by transit ISPs meet the demands of interactive Internet applications, like web browsing and VoIP, which require low end-to-end latency to ensure quick response times. This is reflected in the SLAs between customer and provider ISPs. These SLAs typically specify upper bounds on latency and loss rate of packets transferred within the transit ISP's backbone [Sprb]. The transit ISP guarantees a refund if the observed traffic characteristics exceed the declared upper bounds.

Although large amounts of bulk content are generated and consumed at the edge of the Internet, at the time of writing transit bandwidth remains expensive, with monthly costs varying between $30K and $90K per Gbps depending on the region of the world [LSRS09]. Surprisingly, even though bandwidth is expensive, the utilization of backbone links in transit ISPs is low. The reason is that, in order to honor SLAs, transit ISPs need to avoid congestion and the ensuing packet delays and losses. As a simple way to decrease the chance of congestion, transit ISPs frequently overprovision their network links. If a link has capacity much higher than the average traffic it usually carries, only an unusually high traffic spike can cause congestion. The net result of the practice of link overprovisioning in transit ISPs is that there is a lot of capacity that lies unutilized in today's backbones. Furthermore, network traffic varies considerably over time, exhibiting periodic diurnal and weekly patterns, which creates additional spare capacity during non-peak hours.

The large amounts of unused bandwidth in backbone links presents a unique opportunity for bulk content delivery. Many bulk transfers, such as large movie downloads or file backups, take a long time to complete and are less sensitive to delays, especially at the granularity of individual packets. Unlike interactive traffic, bulk traffic could be delayed to a later point of time, when the network is lightly loaded and there is more spare bandwidth (e.g., at nighttime). Further, unlike interactive traffic, long-running bulk transfers can tolerate occasional periods of network congestion and do not require the packet-level performance guarantees ISPs offer in their SLAs. Finally, bulk transfers are more tolerant to short term link or path failures, and ISPs do not have to overprovision backup paths a priori for quick failure recovery of bulk transfers.

Unfortunately, transit ISPs cannot easily employ a large fraction of the spare capacity in their networks for such low-priority service without risking congestion. Congestion would lead to packet losses and delays in best-effort traffic, thus breaking the SLAs. Moreover, it's unclear what kind of routing transit ISPs should use in order to maximize the utilization of spare resources in their networks.

2.2. Contributions to improve the efficiency of bulk data delivery

In the first part of this thesis, we make the following contributions to improve the cost-efficiency of bulk data delivery in access and transit ISPs:

Traffic shaping of bulk transfers in access ISPs We study the problem of reducing peak utilization in access ISPs by delaying bulk transfers. We identify a new technique that achieves optimal peak reduction and minimal impact on the performance of targeted bulk transfers. However, we also find that such technique results in large global performance losses for bulk flows that traverse multiple traffic shapers deployed by different ISPs. To solve this problem, we propose to break end-to-end bulk transfers into multiple subtransfers, each traversing only one traffic shaper. Between subtransfers, network traffic has to be temporarily stored in the network.

Opportunistic bulk content delivery in transit ISPs We propose and evaluate the potential of NetEx, an opportunistic bulk delivery system that transit ISPs can deploy in their backbones. NetEx delivers bulk data by opportunistically tapping capacity that is not used by current traffic. NetEx is able to allocate a substantial portion of spare network capacity to bulk applications, and at the same time leave current traffic completely unaffected. With NetEx, transit ISPs can tap previously unused bandwidth and potentially market it by offering a traffic class that does not require strict SLAs. We quantify the potential performance of NetEx through detailed simulations driven by data from real commercial and research backbones.

3. Traffic shaping of bulk transfers in access ISPs

ISPs are currently traffic shaping bulk transfers to reduce traffic costs and the risk of network congestion. ISPs typically traffic shape bulk transfers because they account for most of the traffic.

We now present three canonical examples of traffic shaping policies of bulk transfers in use today. We investigate the benefits of these policies and compare them with more sophisticated policies in later sections.

1. **Traffic shaping bulk applications on a per-flow basis** This policy shapes every flow belonging to bulk transfer applications to some fixed bandwidth. For example, Bell Canada revealed that it throttles traffic from P2P file-sharing applications in its broadband access networks to 256 Kbps per flow [Bel]. Traffic shaping applies to flows both in the downstream and in the upstream direction. Bell Canada chose to traffic shape only P2P file-sharing applications because it found that a small number of users of these applications were responsible for a disproportionate fraction of the total network traffic.

2. **Traffic shaping aggregate traffic** Here, traffic shaping is applied to the aggregate traffic produced by multiple network flows. For example, Comcast handles congestion in its access network by throttling users who consume a large portion of their provisioned access bandwidth over a 5-minute time window [Com]. All packets from these users are put in a lower priority traffic class in order to be delayed or dropped before other users' traffic in case of network congestion. Another example of such a policy was deployed at the University of Washington in 2002 [Fri02]. The university started limiting the aggregate bandwidth of all incoming peer-to-peer file-sharing traffic to 20 Mbps to reduce the estimated costs of one million dollars this type of traffic was causing per year.

3. **Traffic shaping only at certain times of the day** This policy is orthogonal to the previous two policies and is typically used in combination with these. An ISP can decide to traffic shape throughout the day or restrict traffic shaping to specific time periods. For example, the University of Washington shapes P2P traffic during the entire day [Fri02], while Bell Canada and Kabel Deutschland announced to only traffic shape during periods of "peak usage", i.e., between 4:30 pm and 2:00 am [Bel, Röt08]. Since many ISPs pay for transit bandwidth based on their peak load, shaping only during peak usage appears to be an effective way to reduce bandwidth costs.

While the above policies are simple to understand, they raise several questions:

1. How effective are the different traffic shaping policies at reducing network congestion and peak network usage?
2. What is the impact of traffic shaping policies on the performance of the targeted network flows?
3. Are there policies that achieve similar or better reduction in bandwidth costs, while penalizing traffic less?

To answer these questions, we first need to define the precise goals of traffic shaping, as well as the metrics with which we evaluate the impact of traffic shaping policies on network traffic.

3.1. Related work

Traffic shaping already plays an important role in ISPs strategies to reduce network congestion and bandwidth costs. For example, some major ISPs were found to block BitTorrent traffic in their networks [DMHG08], and other ISPs are known to throttle bandwidth-intensive applications or users that consume a disproportional large fraction of bandwidth [Bel, Com, Fri02].

Today's networking equipment enables ISPs to deploy even complex traffic shaping policies at line speeds [pacb, Cisb]. This equipment typically supports deep packet inspection, which allows ISPs to identify the traffic of particular applications, and a wide range of traffic shaping and queue management techniques, including token buckets and priority queueing. The queueing and shaping techniques provided by this equipment – and also used in this work – were introduced in the context of quality-of-service and DiffServ [BBC+98]. They were originally developed to provide flow delay guarantees that are better than best-effort. Today, they are still used to give higher precedence to some traffic (e.g., for voice-over-IP traffic), but also to throttle the bandwidth usage of some applications (e.g., file sharing applications).

There is a large body of literature on splitting TCP connections. Typically, these papers focus on improving the performance of TCP connections over wireless (including ad-hoc and cellular) links [BB95, BS97, KKFT02] or over high-latency connections such as satellite links [HK99]. Unlike our approach, they do not stage large amounts of data in the network, but buffer a few packets to gracefully recover from occasional packet loss. Another very popular example of splitting end-to-end connections are the widely used web proxy caches [Squ] aiming at faster download of web content.

To the best of our knowledge, our work is the first study that characterizes the local and global effects of traffic shaping in the Internet. The only related work we are aware of is from Laoutaris et al. [LSRS09], who quantified how much additional "delay tolerant" data (i.e., data that can tolerate delivery delays of hours or days) ISPs could send for free by exploiting 95^{th} percentile billing and diurnal patterns in today's Internet traffic.

To achieve this, they present and evaluate simple end-to-end scheduling policies as well as "store-and-forward" techniques that use storage deployed in the network. They show that it is possible to transfer multiple Tbytes during off-peak times with no additional costs.

There are three main differences between our work and theirs. First, while [LSRS09] aims to send additional (delay-tolerant) data without increasing bandwidth costs for ISPs[1], our work reduces the peak bandwidth usage of ISPs for today's traffic with only moderate impact (i.e., delay) on shaped flows.

Second, the approach presented by Laoutaris et al. requires fine-grained and real-time information about the load of the network for scheduling decisions, and a transport layer that is capable of instantaneously using all available spare bandwidth for the delay tolerant traffic. On the contrary, our traffic shaping policies can be deployed on today's networking equipment.

Third, while the analysis in [LSRS09] uses data that comprises only aggregate network loads, we use flow-level NetFlow traces that enable us to study the behavior of single TCP flows and perform a more detailed and realistic analysis. Thanks to this detailed analysis we could identify global effects of traffic shaping that are related to TCP characteristics and would have escaped an analysis based on traffic aggregates only.

3.2. Goals and potential of traffic shaping

In this section, we identify three goals for traffic shaping policies as deployed by ISPs: minimizing the peak network traffic, minimizing the number of flows targeted by traffic shaping, and minimizing the impact of traffic shaping on flows. We argue that traffic shaping policies should be designed around these goals, and quantify the potential of such policies through an analysis of real-world network traces.

3.2.1. Network traces

In our analysis of traffic shaping performance, we use publicly available NetFlow records collected at the access links of 35 different universities and research institutions. The records contain incoming and outgoing traffic between these universities and the Abilene backbone [ABI]. Even though our traces come from a university environment, we confirmed that the relevant trace characteristics for our analysis (such as diurnal variations and skewness in flow size distribution) are consistent with those observed in several previous studies of commercial Internet traffic [ANB05, AG03].

The NetFlow records were collected during a 1-week period starting on January 1st 2007, and contain durations and sizes of TCP flows. The NetFlow data has two limitations: (1) long flows are broken down into shorter flows (with a maximum duration of 30 minutes), and (2) flows' packets are sampled with a 1% rate. To recover long flows from the NetFlow data, we combine successive flows between the same TCP endpoints into longer flows using the technique employed in [KBFc04]. To account for the sampling

[1] By increasing the average bandwidth usage to nearly the peak usage.

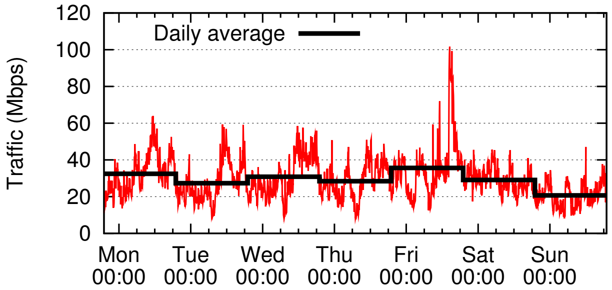


Figure 3.1: Downstream network traffic at Ohio State University: The traffic shows diurnal variations with large peak-to-average ratios.

rate, we multiply packet and byte counts by 100. While this approach is not reliable when applied to small flows, it was shown to be accurate for large bulk flows [ST01b], which are the object of the traffic shaping policies considered in this work.

3.2.2. Goals and potential

We identify the following three goals as the main practical objectives for an ISP that deploys traffic shaping.

Goal 1: Minimizing the peak network traffic. The main motivation for ISPs to deploy traffic shaping is often network congestion [Bel, Röt08]. With traffic shaping, ISPs can lower the risk of congestion by reducing the peak network usage. At the same time, lowering the peak network usage also reduces bandwidth costs for ISPs since they are often charged based on the near-peak utilization (e.g., 95^{th} percentile traffic load) of their links. This creates an incentive for ISPs to keep the peak network usage as low as possible to minimize bandwidth costs. Using our traces, we quantify the maximum peak reduction in network traffic ISPs can achieve with an optimal traffic shaping policy.

Figure 4.1 plots the network traffic in one of our traces (collected at the Ohio State University). The traffic exhibits strong diurnal variations, with traffic peaking around noon and dropping in the early morning. As a result of these variations, the daily traffic peak is considerably higher than the average daily traffic. Intuitively, the lower bound for any realistic peak reduction scheme is the average daily traffic, because this is the minimum traffic level that can assure that all traffic will eventually be delivered within the day[2].

Averaging across all access link traces, the daily peak is 2.6 times larger than the average traffic load, while the 95^{th} percentile is 1.7 times larger than the average traffic. These results suggest that traffic shaping has the potential to reduce ISPs' peak load by a factor of 2.

[2] A higher peak reduction is only possible if traffic is dropped from the network, e.g., by blocking certain applications traffic. However, blocking is a very intrusive form of traffic shaping and ISPs that previously deployed it had to deal with very negative media coverage about this practice [Top07].

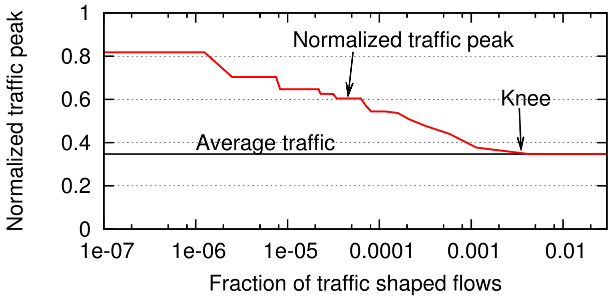


Figure 3.2: Tradeoff between maximum achievable peak reduction and fraction of traffic shaped flows: Intuitively, shaping more flows lowers the peak. However, the peak cannot be lower than the average traffic rate without dropping flows. At this point, shaping more flows has no further benefits.

Goal 2: Minimizing the number of traffic shaped flows. While ISPs have an economic incentive to reduce the peak network usage as much as possible, they are also concerned with affecting as few flows as possible to keep the effect on user traffic low. As a consequence, most ISPs today target either users that are responsible for a disproportional large fraction of traffic (so-called “heavy-hitters”), or applications known to be bandwidth-hungry (e.g., file-sharing applications). Using our traces, we quantify the minimal fraction of bulk flows that need to be shaped to achieve a near-optimal reduction in peak load.

Typically, an ISP would use deep packet inspection to identify flows belonging to bandwidth-intensive applications. However, since our traces do not contain information about application-level protocols, we identify bandwidth-intensive flows based on their size, i.e. the number of transferred bytes.

We sorted all flows in each of our trace by decreasing size. We then selected all flows larger than a certain size T for traffic shaping and computed the theoretical maximum peak reduction achievable. For this analysis, we assume that flows can be arbitrarily throttled, as long as they complete within the trace's time-frame of 1 week. We then repeated this for decreasing values of T, thus selecting more and more flows. Figure 3.2 plots the results for one of our traces. After selecting only 0.4% of the largest flows, the traffic peak reaches the average traffic load and no further reduction is possible (the “knee” in the figure). In this trace, this translates to flows that are larger than 10 MB. Across all traces, traffic shaping less than 4% of the flows is always sufficient to achieve the maximum peak reduction, and in 29 of our 35 traces traffic shaping less than 1% of the flows also suffices. This result suggests that ISPs can considerably reduce their peak while shaping a very small fraction of flows.

Goal 3: Minimizing the delay that traffic shaped flows incur. We found that ISPs have to shape only a small fraction of flows to achieve an optimal reduction in peak network usage. Note that this optimal reduction can be achieved without dropping any flows. Instead, in our analysis, we ensured that all shaped flows complete within the time-frame of the trace. However, even if only a small fraction of flows are affected by traffic shaping, ISPs should try to limit the delay incurred by these flows in order to minimally penalize the applications or users generating the bulk flows. With respect to this goal, focusing on bulk flows has the advantage that these flows, being large, have completion times on the order of minutes, hours or even days. Therefore, they can endure considerable absolute delays without severe damage to their performance. For example, the bulk flows in our trace take on average 3.5 minutes to complete when they are not traffic shaped, suggesting that they can be delayed by seconds without negative effects for applications.

In summary, we found that a traffic shaping policy should not only minimize the peak network traffic, but also affect as few flows as possible and minimize its impact on the shaped flows. In the next section, we compare how well different traffic shaping policies perform relative to these goals.

3.3. Local performance of traffic shaping policies

In this section we analyze how different traffic shaping policies perform based on the three metrics from Section 3.2: the peak reduction, the fraction of shaped flows, and the delay shaped flows incur. As we only consider a single traffic shaper in the network path here, we call this the local performance of traffic shaping policies. In Section 3.4, we analyze the effect of multiple traffic shapers in the networking path.

3.3.1. Selecting flows for traffic shaping

ISPs target only a subset of flows for traffic shaping, typically flows from bandwidth-intensive applications. Doing so, ISPs achieve very good peak reductions while keeping the number of affected flows low. In the following, we call flows that are subject to traffic shaping “low-priority traffic“ and the remaining flows ”best-effort traffic“.

To identify flows from bandwidth-intensive applications, ISPs often employ deep packet inspection (DPI), which is widely available in routers [Cisb] or provided by special DPI equipment [pacb]. Additionally, today’s networking equipment allows ISPs to collect statistics on flow sizes, which can be used to mark large flows for traffic shaping [Cisb, Ipo09]. In practice, flow classification is implemented at ISPs’ ingress routers. Flows are marked as low-priority or best-effort by setting the DSCP field in the IP header[3]. The traffic shaping equipment then selects the packets to traffic shape just based on the value of the DCSP field.

As our traces do not contain information to identify application protocols, we rely on flow sizes instead, i.e., flows that are larger than a certain ”flow size threshold“ are shaped. Picking the right flow size threshold is nontrivial, because a higher threshold will affect fewer flows, but at the same time will give ISPs fewer bytes to traffic shape, and thus limit its ability to decrease peak usage. To select the right threshold for each

[3] The DSCP field allows up to 64 different traffic classes.

trace, we use the analysis from Section 3.2.2 and pick the threshold that results in the maximum potential for peak reduction with the minimum fraction of flows being shaped.

In the traffic shaping policies in this section, unless explicitly stated otherwise, we keep a running counter of the bytes sent by each active network flow, and use its value to classify the flow. For example, if the flow size threshold is 10 MB, a 20 MB flow will send the first 10 MB as best-effort traffic. After that, the flow is classified as low-priority traffic and the remaining 10 MB of the flow are traffic shaped. This technique can also be used by ISPs to deploy a protocol-agnostic traffic shaping policy that targets all flows larger than certain flow size threshold. Note that modern traffic shaping equipment is capable of keeping such per-flow state even on high-speed links [Cisb].

3.3.2. Selecting aggregate bandwidth limits

Some traffic shaping policies (e.g., as used by the University of Washington [Fri02]) shape low-priority flows only when the traffic rate exceeds a certain "bandwidth limit". This limit can refer to the aggregate traffic (best-effort + low-priority traffic) or to the low-priority traffic only. For example, an ISP could traffic shape only when the total traffic rate exceeds 20 Mbps or when the low-priority traffic alone exceeds 20 Mbps.

The bandwidth limit determines the total reduction in traffic peak. As we showed in Section 3.2, the average traffic rate is the minimum value that enables delivery of all traffic. Therefore, in all policies that use a bandwidth limit, we set the bandwidth limit to the average traffic rate of the previous day plus 5% to account for small increases in demand. We found that this approach works well in practice because the average rate is quite stable across days. In fact, in our 35 1-week traces, we found only two days were this was not the case, i.e., the average traffic varied considerably from one day to the next. If there is a sudden increase in daily average traffic, too many low-priority flows may compete for too little bandwidth, thus incurring large delays or even starvation. To overcome this problem, ISPs can monitor the bandwidth of the low-priority flows and of the overall traffic in their network and increase the bandwidth limit if they detect a significant difference from the previous day.

3.3.3. Traffic shaping policies

We now describe the traffic shaping policies we evaluate. All of the traffic shaping policies described here can be implemented using well-known elements like token buckets, class-based rate limiting, and strict priority queuing, available in today's networking equipment [Cisc, Lin]. To design the traffic shaping policies we start from the real-world examples from Section 3 and develop more complex policies, which attempt to reduce the peak traffic while minimize the delay incurred by the traffic shaped flows. Note that all of the traffic shaping policies presented here shape only flows classified as low-priority; best-effort traffic is never shaped.

Per-flow bandwidth limit (PBL) With PBL, each low-priority flow is shaped to a fixed maximum bandwidth. Traffic shapers use a dedicated queue for each low-priority flow, and dequeue packets according to a token bucket algorithm. In our simulations, we limit the bandwidth consumed by each low-priority flow to 250 Kbps.

We also evaluate a variant of this policy called **PBL-PEAK**, where low-priority flows are shaped only between 9 am and 3 pm local time. This period corresponds to 6 hours centered around the peak utilization in our traces at about noon. Both PBL and PBL-PEAK require routers to allocate a new queue for each new low-priority flow, thus potentially limiting the practicality of these two policies.

Low-priority bandwidth limit (LBL) In this policy, the aggregate bandwidth consumed by all low-priority flows is bound by a bandwidth limit. Traffic shapers deploy two queues: one for best-effort traffic and one for low-priority traffic. A token bucket applied to the low-priority queue limits the low-priority traffic rate to the desired bandwidth limit. The bandwidth limit is determined based on the average bandwidth consumed by low-priority traffic on the previous day, as described before. No bandwidth limit is applied to the best-effort traffic. This policy can also be used to approximate PBL by using a dynamic bandwidth limit proportional to the number of low-priority flows.

Aggregate bandwidth limit (ABL) When the aggregate traffic (best-effort + low-priority traffic) approaches the bandwidth limit, low-priority flows are shaped to keep the aggregate traffic below the limit. Note that best-effort traffic is never shaped. Therefore, if the best-effort traffic exceeds the bandwidth limit, this policy cannot guarantee that the aggregate traffic stays below the bandwidth limit. However, in such cases the traffic shaper throttles the low-priority traffic to zero bandwidth until the best-effort traffic falls below the bandwidth limit.

To implement this policy, traffic shapers deploy two queues: a high-priority queue for the best-effort traffic and a low priority queue for the low-priority traffic. Both queues share a single token bucket, which generates tokens at a rate corresponding to the aggregate bandwidth limit. Each time a packet from either queue is forwarded, tokens are consumed. However, best-effort packets are always granted access to the link, even if there are not enough tokens left. This is unlike an ordinary token bucket and can cause the token count to occasionally become negative, thus precluding low-priority packets from using the link. As long as the total traffic rate is below the bandwidth limit, there are always enough tokens to forward both best-effort and low-priority traffic. But, as the total traffic level exceeds the bandwidth limit, low-priority flows are shaped.

Aggregate bandwidth limit with shortest-flow first scheduling (ABL-SFF) This policy is as ABL, but additionally optimizes the usage of the bandwidth available to the low-priority flows. Unlike PBL or LBL, in ABL low-priority traffic is not guaranteed a minimum bandwidth allocation, but all low-priority flows compete for the bandwidth the best-effort traffic is not using. Thus, when the total traffic reaches the bandwidth limit, the bandwidth available to low-priority flows becomes so low that some of these flows get substantially delayed or even stalled.

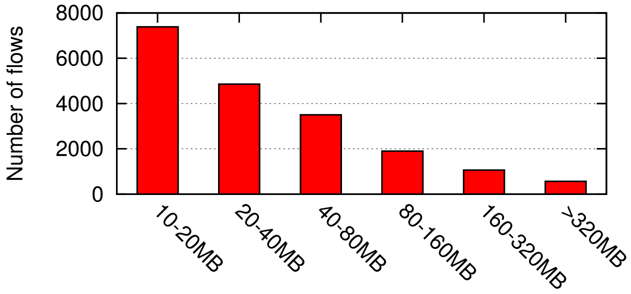


Figure 3.3: Number of flows >10 MB per flow size range for the Ohio State trace.

We gained an insight on how to lessen this problem by looking at the flow-size distribution in our traces. Figure 3.3 shows the number of low-priority flows that fall into different size ranges in one of our traces. The distribution of flow sizes is heavily skewed with roughly 85% of low-priority flows having size between 10 MB and 100 MB. Such a flow size distribution is quite common in Internet traffic. [SRS99]. Under such skewed distributions, it is well-known that giving priority to small flows reduces the mean completion time [GM02, MG00]. Therefore, in the ABL-SFF policy, when selecting a low-priority packet to send over the link, the traffic shaper always chooses the packet from the flow with the smallest size. This effectively replaces the usual FIFO queueing with shortest-flow-first queueing. To implement this policy, the traffic shaper needs to allocate a separate queue for each low-priority flow. Also, the shaper needs a priori knowledge of the size of each flow to select the next low priority packet. This makes this policy not directly applicable to general network flows, whose size cannot be known, but gives an useful lower-bound on the minimum delay that low-priority flows incur with the ABL policy.

Aggregate bandwidth limit with strict priority queuing (ABL-PQ) This policy is a practical version of ABL-SFF and can be implemented by ISPs with today's equipment. It approximates the shortest flow first scheduling of ABL-SFF as follows. First, unlike ABL-SFF, it does not assume a priory knowledge of flow sizes, but instead keeps a running count of the bytes sent by each active network flow and uses this value as an estimate of the flow size. Second, ABL-PQ does not use a separate queue for each low-priority flow, but instead uses a fixed, small number of low-priority packet queues. Each queue accommodates packets of low-priority flows whose size fall in a given range. When the traffic shaper has bandwidth to send low-priority traffic, it schedules the low-priority queues giving *strict priority* to the queues that accommodate smaller flows.

To balance the load of the low-priority queues, we selected contiguous ranges of exponentially increasing width. This is motivated by the typical skewness of the flow size distribution in the Internet. For our traces, where flows larger than 10 MB are classified as low-priority traffic, the first low-priority queue contains packets of flows that have transferred between 10 MB and 20 MB, the second queue contains packets of flows that

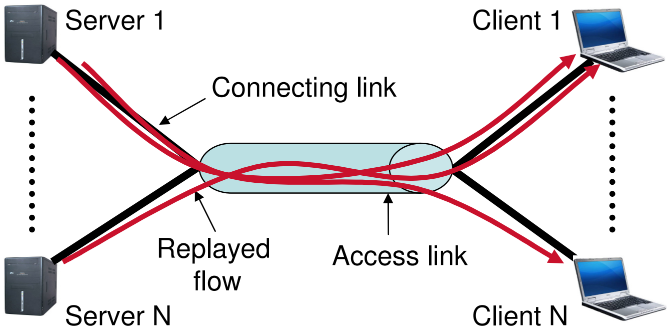


Figure 3.4: Simulation topology: All replayed TCP flows cross a shared access link where traffic shaping takes place.

have transferred between 20 MB and 40 MB, and so on. As opposed to ABL-SFF, this policy uses a limited number of queues (we use 6 in our experiments) and can be easily supported by today's networking equipment. Remember that ISPs typically deploy flow classification at their ingress points and that network equipment is capable of keeping per-flow state [Cisb, Ipo09].

3.3.4. Comparison methodology

We used trace-driven simulations to study the behavior of flows under various traffic shaping mechanisms. We conducted our analysis using the ns-2 simulator and the traces from Section 3.2. During the simulation, we replayed all TCP flows in each trace using the ns-2 implementation of TCP-Reno.

We used the simulation topology shown in Figure 3.4 to analyze traffic shaping over an access link. We faced an interesting challenge while replaying the TCP flows: our traces included information about flow arrival times, sizes, and durations, but we lacked information about flow round-trip times (RTTs) and loss rates. To simulate packet losses, we set the capacity of the link connecting the server node for each flow to match the average bandwidth of the flow (see Figure 3.4). This ensures that the simulated flows complete in similar durations as the original flows in the trace. Furthermore, we picked the RTT of a flow choosing from a distribution of latency measurements using the King tool [GSG02]. We found that the aggregate bandwidth of the simulated flows match the one of the original flows from the traces very well.

To compare different traffic shaping policies, we focused on the three metrics from Section 3.2: the achieved peak reduction, the fraction of shaped flows, and the delay shaped flows incur.

3.3.5. Results

We now present the results of the comparison of the different traffic shaping policies.

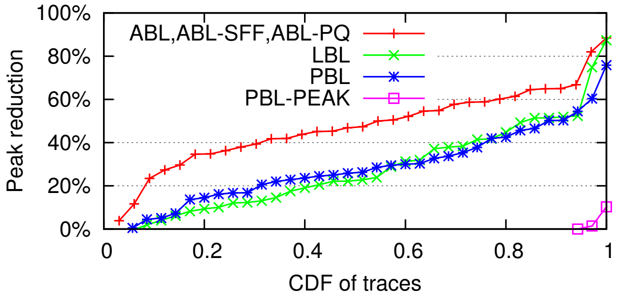


Figure 3.5: Reduction in peak with different traffic shaping policies: Traffic shaping policies based on aggregate bandwidth limits (ABL) achieve considerable reductions in peak traffic.

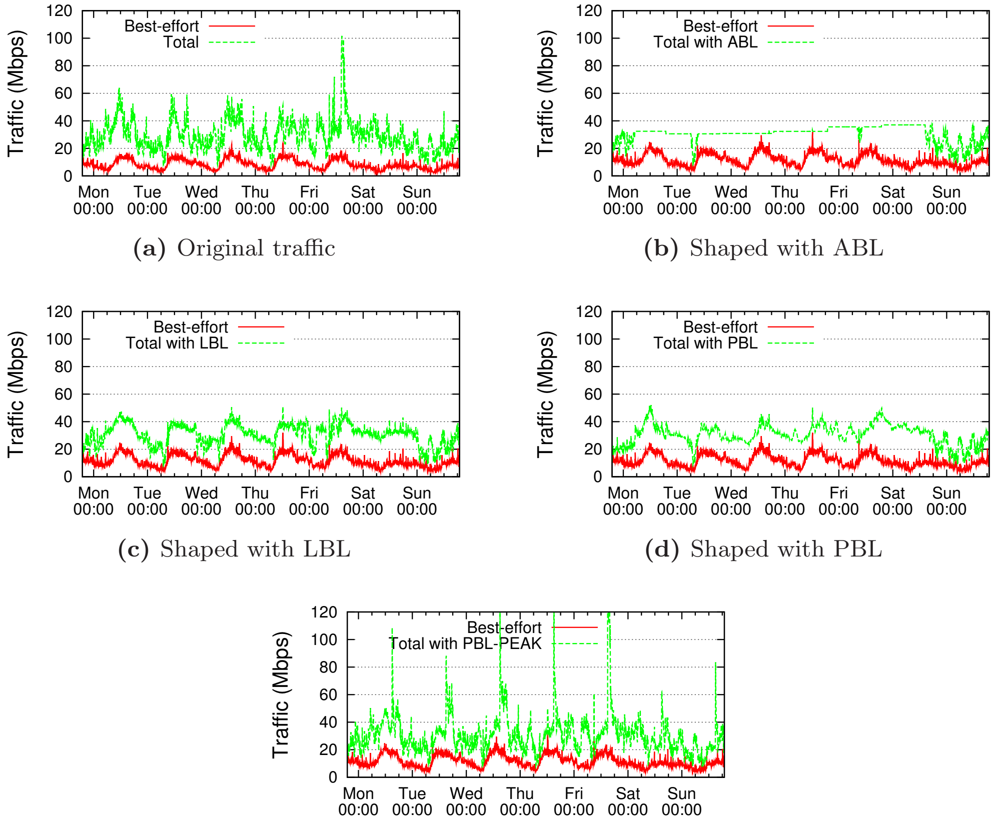
Peak reduction

We start by presenting the overall peak reductions attained by the different policies across all our traces, shown in Figure 3.5. Since ABL, ABL-SFF and ABL-PQ all cap the traffic at the same limit, we report only one line for all of them. The ABL policies achieve a considerably higher peak reduction than LBL. This is because LBL does not take into account the level of best-effort traffic when computing the low-priority traffic cap. PBL performs similarly to LBL, while PBL-PEAK is by far the worst-performing policy, causing in 90% of the cases an *increase* in traffic peak (these correspond to points that lie on the negative side of the y-axis in the figure, and are not shown).

To better understand the differences in peak reduction among the different policies, we show in Figure 3.6 time plots of the traffic in an example trace. Flows smaller than 10 MB are marked as best-effort traffic. Figure 3.6(a) shows the original traffic trace without traffic shaping. Compared to the original trace, the ABL policies (Figure 3.6(b)) considerably reduce peak bandwidth (-64%). LBL (Figure 3.6(c)) achieves lower, but still substantial reductions (-51%).

Comparing LBL and ABL, we observe that ABL achieves a much smoother peak as the total amount of traffic is capped to a constant daily threshold (note that best-effort traffic can still occasionally exceed the threshold). The advantage of LBL is that it guarantees a minimum amount of bandwidth to low-priority traffic, and thus avoids stalling low-priority flows. However, the total traffic still shows diurnal patterns and the peak reduction is thus not as a large as with ABL.

Finally, Figure 3.6(d) and 3.6(e) show PBL and PBL-PEAK, respectively. Interestingly, PBL-PEAK is largely ineffective at reducing traffic peak. In fact, PBL-PEAK increases the traffic peak by 11% in this case. To understand this counterintuitive result, consider the following example. During the traffic shaping period (9 am to 3 pm), each low-priority flow is throttled to 250 Kbps. This small per-flow bandwidth makes it hard for low-priority flows to complete. As a result, the number of active low-priority flows increases during the traffic shaping period. At the end of the traffic shaping period all these flows are given full bandwidth again, which they promptly consume. This causes



(a) Original traffic

(b) Shaped with ABL

(c) Shaped with LBL

(d) Shaped with PBL

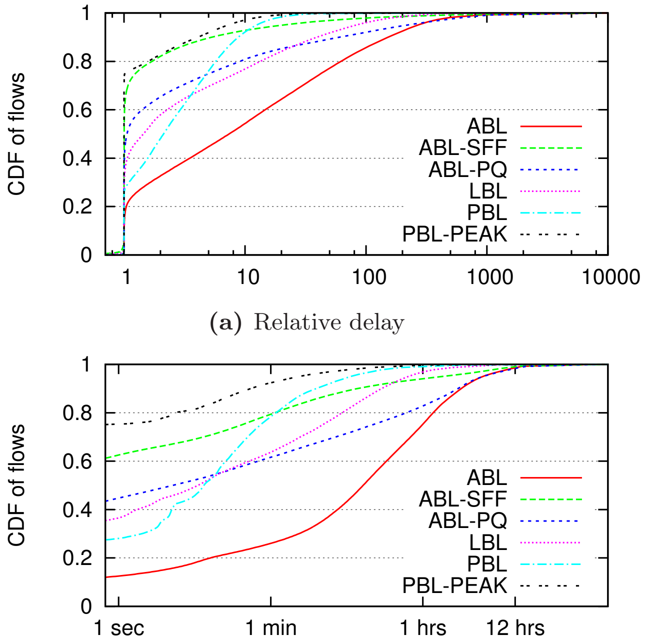
(e) Shaped with PBL-PEAK

Figure 3.6: Traffic in the Ohio State trace with different traffic shaping policies: Each plot shows best-effort traffic as well as the total amount of traffic (best-effort + low-priority traffic).

the traffic spikes that are visible in Figure 3.6(e) on each day at 3 pm, i.e., the end of the traffic shaping period. These spikes can be considerably higher than the original traffic peak. This phenomenon does not occur with PBL because traffic shaping occurs throughout the day.

Number of delayed low-priority flows

Since in our analysis all traffic shaping policies use the same flow size threshold, the flows that are treated as low-priority by each traffic shaping policy are the same. However, depending on the policy, some of these flows may incur only moderate delay. We regard a low-priority flow as delayed if its completion time increases by more than 5% compared to when no traffic shaping is in place. Table 3.1 reports, across all traces, the fraction of low-priority flows that are delayed by more than 5% with each traffic shaping policy and the achieved average peak reduction. ABL affects the most flows, followed by PBL,



(a) Relative delay

(b) Absolute delay

Figure 3.7: CDFs of relative and absolute delays for low-priority flows across all our experiments: The relative delay is the ratio of the completion time of the traffic shaped flow to its completion time with no traffic shaping. With the exception of ABL and ABL-PQ, few low-priority flows get delayed by more than one hour, and almost none is delayed by more than 12 hours.

which only gives 250 Kbps to each flow. Compared to ABL, ABL-SFF and ABL-PQ greatly reduce the number of delayed flows. PBL-PEAK delays very few flows because it only rate limits for 6 hours a day, but it also significantly increases the peak usage as pointed out above. Interestingly, although LBL always allocates a minimum amount of bandwidth to low-priority flows, it delays more flows than ABL-PQ and ABL-SFF, which do not provide such a guarantee. The reason is that both ABL-PQ and ABL-SFF give priority to smaller flows, thus shifting the bulk of the delay to a few large flows.

Delay of low-priority flows

Figure 3.7 plots the CDFs of relative and absolute delays of low-priority flows for different policies across all our experiments. ABL causes the largest delays while both ABL-SFF and PBL-PEAK lead to very low delays. However, as mentioned above, PBL-PEAK also significantly increases peak usage and has therefore little utility. With ABL, about half of low-priority flows take 10 times longer or more to complete compared to when they are not traffic shaped. With ABL-PQ, only 20% of low-priority flows take 10 times longer or more to complete. Regarding the absolute delay of flows (Figure 3.7(b)), we

Policy	Flows delayed by >5%	Average peak reduction
ABL	80%	48%
PBL	71%	29%
LBL	61%	28%
ABL-PQ	51%	48%
ABL-SFF	32%	48%
PBL-PEAK	24%	-87%

Table 3.1: Fraction of low-priority flows delayed by more than 5% and average peak reduction: Among the practical policies that maximize peak reduction, ABL-PQ delays the fewest flows.

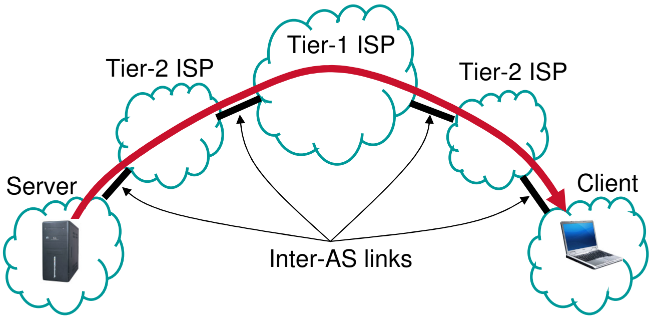


Figure 3.8: Flow traversing multiple ISPs: It is likely that a transfer between a server and a client traverses multiple traffic-shaping ISPs.

observed that at most 20% of low-priority flows are delayed by more than 1 hour for all policies, and almost no flow is delayed by more than 12 hours.

3.3.6. Summary

We compared the performance of 5 traffic shaping policies with respect to our goals of peak reduction, minimum number of delayed flows, and minimum increase in completion time. We found that the ABL policies result in the best peak reduction (almost 50% in half of our traces). In addition, ABL-SFF keeps the delay incurred by low-priority flows to a minimum. However, it might not be possible to implemented ABL-SFF in practice as it requires a distinct router queue for each low-priority flow. A more practical alternative to ABL-SFF is ABL-PQ, which achieves both high peak reduction and moderate delay of low-priority flows.

3.4. The global impact of local traffic shaping

In this section, we focus on the impact wide-spread deployment of traffic shaping has on the end-to-end performance of bulk flows in the Internet. As economic incentives are

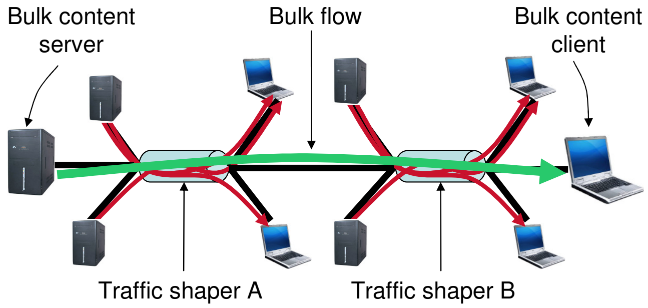


Figure 3.9: Simulation topology for analyzing the performance of a flow passing two traffic shapers: A long-running bulk TCP flow transfers data from the server to the client traversing two traffic shapers that act independently.

likely to drive ISPs to deploy traffic shapers at the boundaries of their networks, long flows may be subject to traffic shaping at multiple inter-AS links (see Figure 3.8).

Our goal is to understand how bulk transfers are affected by multiple independent traffic shapers along their paths. This is in contrast to our analysis in the previous section that analyzed the behavior of flows passing through a single traffic shaper.

For the analysis, we assume that each traffic shaper implements the ABL-PQ policy from the previous section, as this policy enables maximum peak reduction with low impact on network flows.

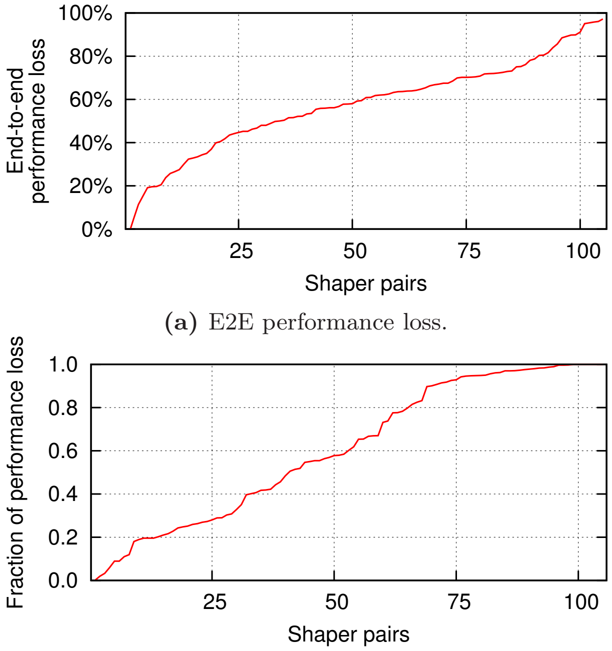
3.4.1. Analysis methodology

Our analysis is based on trace-driven simulation experiments conducted using ns-2. Figure 3.9 shows the topology we used in our analysis; it consists of two traffic shaped links connected to each other. We used our university traces to simulate the local traffic traversing each of the shapers, according to the methodology we described in Section 3.3.4. In addition to the flows from the traces, we simulated a week-long bulk TCP flow that traverses both traffic shaped links. We analyzed the performance of this week-long bulk flow to understand the impact of multiple traffic shapers. We focused on a single long-running bulk flow because small flows are left largely unaffected by the ABL-PQ policy.

We also ran simulation experiments with the long running bulk flow traversing each of the traffic shaped links separately. This allows us to compare the flow's performance when it is crossing a pair of traffic shapers with the flow's performance when it is crossing each traffic shaper separately.

Although our long-running flow is active throughout the simulated week, we focus solely on the performance achieved from Tuesday to Thursday. The reason is that there is often sufficient available bandwidth to serve all traffic around weekends, and as a consequence our traffic shapers are mostly active during the central days of the week.

As a measure of a bulk flow's performance, we count the number of bytes the bulk flow was able to send from Tuesday to Thursday. To quantify the impact of multiple



(a) E2E performance loss.

(b) Fraction of loss due to offsets in the shapers' available bandwidth.

Figure 3.10: The impact of two traffic shapers on the path: The E2E performance loss is significant in most cases (a). In only a few cases can the loss be entirely attributed to offsets in the shapers' available bandwidth (b).

traffic shapers on a flow, we define a metric called *end-to-end performance loss*. End-to-end performance loss is defined as the relative decrease in performance of a bulk flow traversing multiple traffic shapers compared to the minimum performance the bulk flow achieves when it traverses either of the two traffic shapers separately. More formally, consider a flow that transfers B_1 and B_2 bytes when it separately traverses traffic shapers S_1 and S_2, respectively. If the same flow transfers G bytes when it simultaneously traverses S_1 and S_2, the end-to-end performance loss of the flow is: $(min(B_1, B_2) - G)/min(B_1, B_2)$.

3.4.2. The impact of multiple traffic shapers on end-to-end performance

To study the effects of multiple traffic shapers, we used traces from 15 of our 35 university access links. We simulated traffic shaping on these links and analyzed the performance of bulk flows over the all possible (105) pairings of the 15 traffic shaped links. The universities are spread across four time zones in the US. When replaying the traces in simulation we adjusted for the differences in local time by time shifting all traces to the Eastern Standard Time. We discuss the impact of the differences in local time zones of traffic shapers in the next section.

Figure 3.10(a) shows the end-to-end performance loss experienced by flows traversing pairs of traffic shaped links relative to their performance when they cross each of the traffic shaped links individually. Even when the traffic shapers are in the same time zone, the loss in end-to-end performance is significant. In almost 80% of the cases, flows crossing two shapers sent 40% less data than what they would have sent over either of the shapers independently. In 50% of the pairings, the loss in performance is larger than 60%. While we do not show the data here, the performance continues to slide dramatically for each additional traffic shaper.

Factors affecting end-to-end performance

The considerable performance loss for flows traversing multiple traffic shapers can be mainly attributed to two factors. First, at any time t of the simulation, a flow traversing two traffic shapers S_1 and S_2 is limited to using only the *minimum bandwidth available at either traffic shaper at time t*. However, the sum of these minima over the entire simulation time can be lower than the total bandwidth available at either of the two traffic shapers during the same time. More formally, if T is the total simulation time and B_i^t is the bandwidth available at time t at traffic shaper S_i, then $\sum_{t=1}^{T} B_i^t$ is the total bandwidth available at each traffic shaper S_i during T. The first limiting factor can therefore be written as:

$$\sum_{t=1}^{T} min(B_1^t, B_2^t) \leq min(\sum_{t=1}^{T} B_1^t, \sum_{t=1}^{T} B_2^t) \tag{3.1}$$

We refer to the loss in performance due to this factor as the *loss due to offsets in shapers' available bandwidth*. This loss is visually explained in Figure 3.11, which shows a network path that traverses two shaped links. Figures 3.11(a) and (b) show how the bandwidth available at each shaper varies over time. A flow that traverses both shapers is limited by the minimum bandwidth available at either shaper, which is shown in Figure 3.11(c). As a consequence, the flow can use only a fraction of the bandwidth available when it traverses either traffic shaper in isolation.

The second limiting factor is that TCP congestion control may prevent the flow from fully using the bandwidth available at any time t. Because each traffic shaper is throttling bulk flows independent of other shapers, multiple traffic shapers can lead to multiple congested (lossy) links along an Internet path. A long TCP flow traversing two or more congested shapers would be at a serious disadvantage when it competes for bandwidth against shorter flows traversing only a single shaper. Prior studies [Flo91] have shown that multiple congested gateways can lead to an additional drop in the end-to-end performance of a TCP flow. We refer to this performance loss as the *loss due to TCP behavior*.

Estimating the impact of the factors

Next we investigate the role of the two factors, namely, *loss due to offsets in shapers' available bandwidth* and *loss due to TCP behavior*, in the drop in end-to-end performance

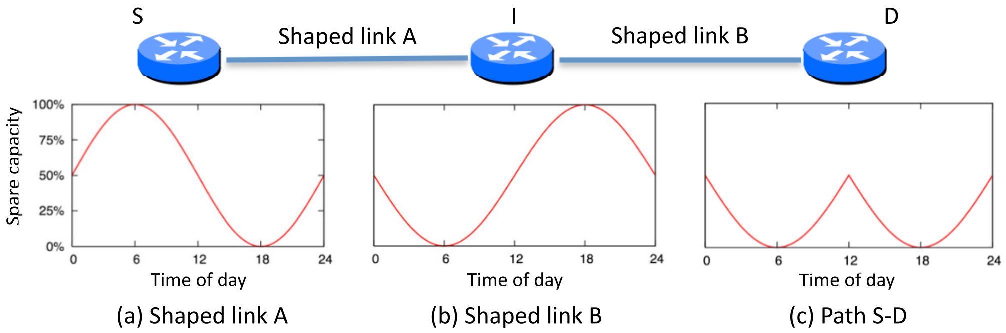


Figure 3.11: Transfer from a source S to a destination D through two shaped links: at any time, transfers are limited to using the end-to-end bandwidth available along the $S-D$ path, that is, the minimum of bandwidths available on links A and B.

of bulk flows. To estimate the performance drop due to offsets in available bandwidth, we do the following: for each pair of traffic shapers, we compute the number of bytes that could have been transferred by a hypothetical flow that fully uses the minimum bandwidth available at either of the shapers at all times during the simulation. In other words, the performance of the hypothetical flow is not affected by the second factor, i.e., TCP behavior when crossing multiple congested shapers.

Figure 3.10(b) shows the fraction of end-to-end performance loss that can be attributed to offsets in the shapers' available bandwidth. The bandwidth offset between shapers accounts for the entire performance loss only in a few cases, while in many cases it accounts for less than half of the performance loss. We attribute the remaining performance loss to the penalty TCP flows suffer when traversing multiple bottlenecks. Simultaneously competing with other flows at multiple congested links takes a heavy toll on the overall performance of a TCP flow, and traversing multiple traffic shapers makes this scenario very likely.

3.4.3. The impact of traffic shaping across time zones

If multiple traffic shapers are in the same time zone they also share similar night and day cycles. However, if they are many time zones apart from each other, their night and day time cycles will be out of phase. This can cause a severe decrease in the end-to-end performance of passing bulk flows. In the previous section, we found that the end-to-end performance drops significantly even when the shapers are in the same time zone. Next, we investigate whether we will see additional loss in performance when these traffic shapers are separated by additional time zones. We repeat the simulations

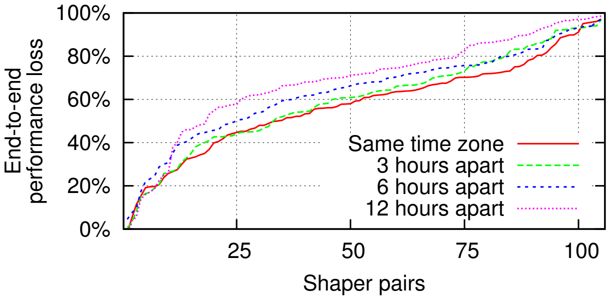


Figure 3.12: Impact of time zone offset on performance: Compared to two traffic shapers located in the same time zone, topologies with 3, 6, and 12 hours time difference in the location of the shapers loose performance only moderately.

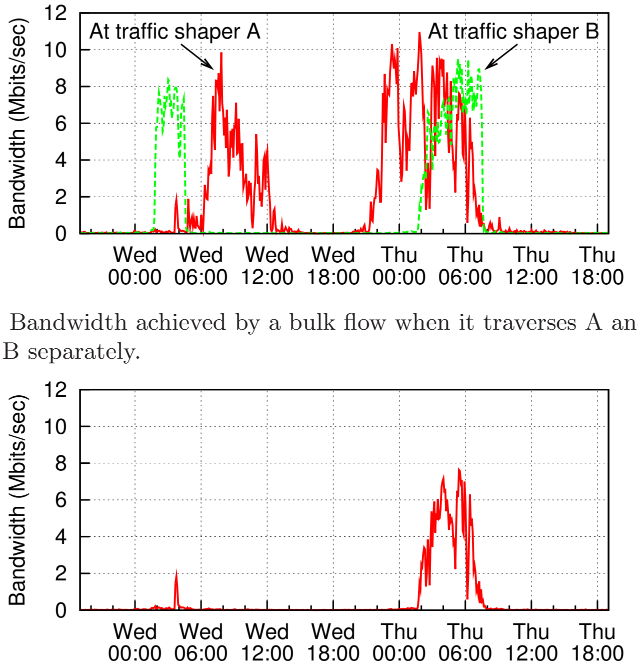
Time zone difference	**Median total perf. loss**	**Avg. total perf. loss**
0 hrs	60%	58%
3 hrs	62%	60%
6 hrs	68%	64%
12 hrs	72%	69%

Table 3.2: Median and average loss in performance for different time zone offsets: Compared to the loss in performance caused by adding a second traffic shaper to the path, increasing the time difference of this shapers increases the loss only moderately.

with two traffic shapers, but vary the time zone difference between the two shapers by time-shifting the traces before they are replayed.

In Figure 3.12, we plot the end-to-end performance loss for simulations where the two shapers are in the same time zone and for simulations where the two shapers are 3, 6, and 12 hours apart. We show the average performance loss in Table 3.2. Strikingly, while the performance loss does increase with the time zone difference between traffic shapers, the additional loss is rather small compared to the performance loss incurred when the two shapers are in the same time zone. While two shapers in the same time zone result on average in 58% loss in performance, spacing them by 3 hours decreases performance only by an additional 2%. A time shift of 12 hours results in an average total performance loss of 69%.

To understand why most performance loss is suffered when two traffic shapers are in the same time zone, we took a closer look at the bandwidth achieved by low-priority flows when they traverse only one traffic shaper. Figure 3.13(a) plots the bandwidth achieved by a long bulk flow over the course of two days when it traverses two shapers A and B in isolation. Interestingly, the flow exhibits diurnal patterns with a very high peak-to-trough ratio; it reaches a very high peak throughput, but only for a short time between



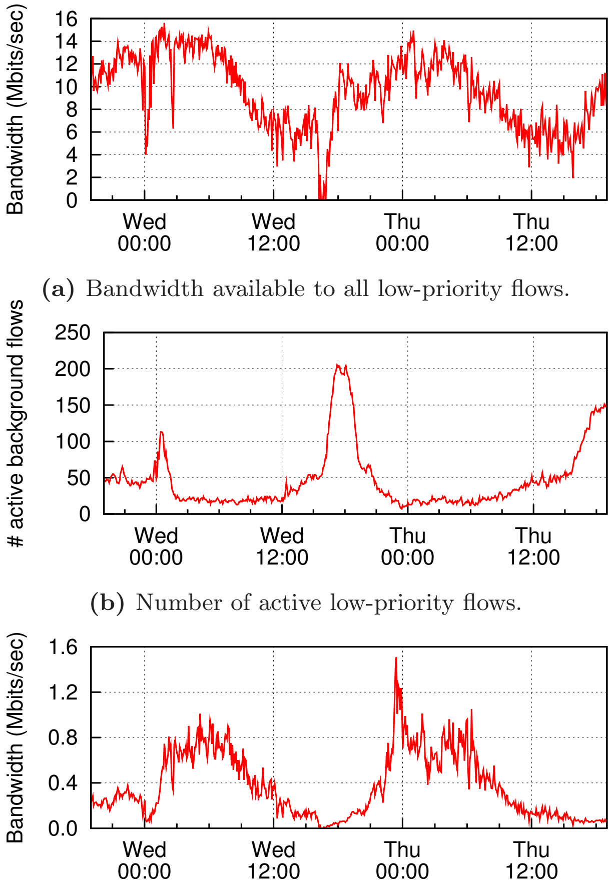
(a) Bandwidth achieved by a bulk flow when it traverses A and B separately.

(b) Available bandwidth when the bulk flow traverses both A and B.

Figure 3.13: Traffic shaped TCP transfers are extremely bursty: When a flow traverses one traffic shaper, most bytes are transferred within a small amount of time, i.e., within the peak periods in (a). When a flow traverses multiple shapers, and the peak periods across these shapers do not overlap nicely, the flow incurs a high loss in E2E performance (b).

midnight and early morning. In fact, we found that more than 90% of all bytes are transferred during less than 10% of the flow's total duration. Thus, when a flow crosses both traffic shapers A and B, even a marginal misalignment in the peak throughput periods of A and B can lead to a dramatic drop in end-to-end throughput. We show this in Figure 3.13(b), which plots the bandwidth available on a path traversing both shapers A and B. Such small misalignments in the peak throughput periods can occur even when shapers are in the same time zone. This explains why time zone differences between traffic shapers result in a relatively small additional loss in end-to-end performance.

The extreme diurnal patterns exhibited by a single traffic shaped bulk flow stands in contrast to the more gentle diurnal patterns exhibited by the aggregate bandwidth available to all bulk flows. We explain the reasons for the difference in Figure 3.14. Figure 3.14(a) plots the total bandwidth available to all low-priority flows at traffic shaper A over time. The peak-to-average ratio in available bandwidth is approximately two, consistent with our observations in Section 3.2.2. Figure 3.14(b) plots the number of active low-priority flows that compete for this bandwidth. The number of active



(a) Bandwidth available to all low-priority flows.

(b) Number of active low-priority flows.

(c) Bandwidth available per flow.

Figure 3.14: Traffic shaping can lead to very bursty transfers: The drops in available bandwidth to low-priority flows over time (a) cause the number of active TCP flows to increase when bandwidth is scarce (b), leading to much sharper peak to trough ratios in the per-flow available bandwidth (c).

flows increases sharply at times when the available bandwidth is low, because new flows arrive at the traffic shaper and existing ones don't complete due to lack of bandwidth. The number of active flows decreases sharply again when more bandwidth becomes available, resulting in very pronounced diurnal patterns in the number of active flows. Figure 3.14(c) plots the fair share of bandwidth for each bulk flow, which is obtained by dividing the aggregate available bandwidth by the number of active flows at any point in time. The per-flow bandwidth exhibits considerably sharper diurnal patterns than the aggregate bandwidth due to the variation in number of active flows over time. This explains why traffic shaped flows transfer most of their bytes during a short window of time in which they achieve their peak throughput.

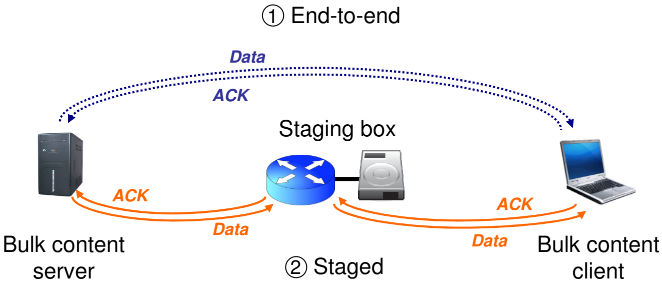


Figure 3.15: Simulation topology used to evaluate the staging service: A staging box is deployed on the network path between the 2 traffic shapers, breaking the bulk transfer into two independent subtransfers.

3.4.4. Summary

In this section, we identified two main factors that affect the performance of long bulk flows traversing multiple traffic shapers: *loss in end-to-end bandwidth* and *loss due to TCP behavior*. First, we found that a long bulk flow traversing two traffic shapers suffers a considerable loss in performance, and in many cases the expected loss in available end-to-end bandwidth is not enough to warrant such a high loss. In these cases, most of the loss in performance comes from the unfairness suffered by a TCP flow when it traverses multiple congested links. Second, we found that there is no large additional loss when the two traffic shapers are located in different time zones. The reason for this is that a long bulk flow traversing a single traffic shaper transfers *most of its data in a short time window*. Thus, when a flow traverses multiple traffic shapers, its end-to-end performance depends on how well the time windows at each shaper overlap. However, because these time windows are short, there is a high chance that they poorly overlap even when the traffic shapers are in the same time zone.

3.5. Improving the performance of bulk transfers with staging

In the previous sections, we showed that ISPs have an incentive to selfishly traffic shape interdomain bulk transfers to reduce their transit costs. However, as more and more ISPs do that, the end-to-end performance of bulk transfers is likely to decrease dramatically, directly affecting the ISPs' customers. In this section, we investigate whether it is possible to avoid such a tragedy of the commons, without restricting ISPs from deploying locally-optimal traffic shaping policies.

3.5.1. Isolating the effects of local traffic shaping with staging

The root cause of the global slowdown of bulk transfers is the harmful interaction between traffic shapers local to different ISPs. Individually, each local traffic shaper affects bulk flows only minimally. But, taken together, multiple traffic shapers along an end-to-end path inflict substantial performance penalty.

To prevent traffic shapers at different links along a path from interfering with one another, we propose to *stage* bulk transfers. By staging we refer to breaking up an end-to-end transfer along a path into a series of sub-transfers along segments of the path, where each path segment contains only one traffic shaper (see Figure 3.15). The end points of each sub-transfer maintain their own congestion control state, thus isolating traffic shaping within one path segment from affecting the transfers in other segments.

When transfers along different segments are decoupled, data might arrive at a router connecting two successive segments faster along the upstream segment than it can be sent along the downstream segment. In such cases, we need to temporarily store data at the intermediate router connecting the two segments. The buffered data would be drained at a later point in time when there is more available bandwidth on the downstream segment than upstream segment (see Figure 3.11). Thus, staging needs storage at intermediate points to exploit the bandwidth available on each of the upstream and downstream path segments separately and efficiently. In contrast, end-to-end transfers are limited to the minimum bandwidth available across all the path segments at all times.

The amount of storage available at intermediate points crucially determines how effectively staging works. If there is too little storage, it would be hard to overcome large offsets in the times when bandwidth is available across different path segments. Once a router runs out of storage, the transfer on its upstream path segment gets throttled down to the bandwidth available on its downstream path segment, and staging yields no further benefits. On the other hand, adding more storage beyond a certain limit is wasteful and does not improve the performance of the transfers. Our evaluation in section 3.5.3 quantifies the benefits of staging as a function of storage deployed.

3.5.2. Design alternatives

The basic idea behind staging – splitting an end-to-end transport connection into a series of cascading transport connections – has been previously used in other contexts, such as caching web content with proxy servers [FGea99, Apa, Squ], and improving TCP performance over paths with wireless or satellite links as their last hop [BB95, BS97, HK99]. Tremendous research and development have gone into addressing transport layer issues (e.g., end-to-end reliability) that arise when implementing staged transfers [KKFT02]. Rather than reinvent the wheel here, we present a high-level overview of the design alternatives for staging and cite prior work for the details of the design. However, we do discuss the tradeoffs between the designs in terms of their applicability to bulk transfers, their deployment barriers and their deployment incentives.

Proxy server based staging

Our first design involves using popular HTTP proxies [Apa, Squ] for staging bulk content. When a client wants to download bulk content from a server, it simply establishes a transport connection (e.g., TCP) to a proxy and requests content from it. The proxy in turn connects to the server, fetches the content, and forwards it to the client. Thus, the data transfer is staged at the proxy. Note that proxy server itself can connect to another upstream proxy server and establish a chain of inter-proxy transport connections before eventually connecting to the content server. In this case, the bulk transfer would be staged at each of the multiple proxy servers along the path.

For the design to work efficiently, one would have to both deploy proxies at key locations in the Internet and select the proxies such that each segment of the staged transfer contains only one traffic shaper. The proxies could be deployed by the ISPs themselves or by content delivery networks like Akamai [Aka]. Transit ISPs might be encouraged to deploy such proxies because they are inexpensive and offer significant performance benefits to their customers (see Section 3.5.3). Further, there are incremental benefits even in a partial deployment scenario; when a large transit ISP deploys one or more proxies within its backbone network, it immediately benefits transfers between any two of its traffic shaping customers. On the other hand, CDNs like Akamai might be able to leverage their distributed caches world-wide to offer staging service for end users wishing to speed up their bulk downloads at a price.

One disadvantage with the proxy-based approach is that it is non-transparent; clients need to be configured with the address of their upstream proxy. Another potential problem is that only bulk transfers conducted using the HTTP protocol can be staged. This might not be a serious limitation in the Internet today because a large number number of content transfers work over HTTP [Gnu, Fas].

Split-TCP based staging

Our second design is inspired by the Split-TCP designs that are deployed by satellite or cellular broadband ISPs on their last hop [BB95, BS97]. To implement Split-TCP, ISPs deploy boxes (sometimes referred to as Performance Enhancing Proxies or PEPs [MDea00]) that split TCP connections along a path by intercepting data packets from the sender and impersonating the receiver by ACKing the receipt of the packets even before forwarding the packets to the receiver. Simultaneously, the boxes impersonate the sender by forwarding the data packets to the receiver with spoofed source address. Effectively, the bulk transfer is staged at the Split-TCP box.

To stage transfers with Split-TCP, ISPs need to deploy Split-TCP boxes at some intermediate point along the paths taken by the transfers. A transit ISP could deploy such boxes at its customers' access routers, where they can intercept and split all bulk flows to and from its customers.

Compared to the HTTP proxy based approach, Split TCP has two primary advantages. First, it is transparent to end hosts; clients do not need to be configured with addresses of Split-TCP boxes as they are deployed along the network paths by ISPs and intercept the packets automatically. Second, because Split-TCP operates at the transport layer, it works with a wider variety of bulk flows, including those that do not use the HTTP protocol. However, Split-TCP is known to break the end-to-end semantics of TCP (e.g., end-to-end reliability), and there has been significant work and several RFCs devoted to analyzing the resulting risks and potential fixes [MDea00, BKea01, BPea02]. More recently, a number of research efforts have developed variants of Split-TCP that

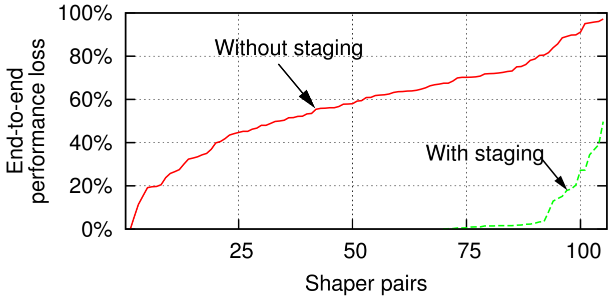


Figure 3.16: Performance loss with and without the staging service: In roughly 90% of the cases, deploying the staging service recovers the full bulk transfer performance.

maintain end-to-end semantics [KKFT02]. However, they require modifications to the TCP implementations at the end hosts.

Summary

Our discussion above suggests that there are many alternative staging designs that could be deployed. Some of the designs are transparent, while others are not. Some can be deployed only by ISPs, while others could be deployed by CDNs like Akamai as well. However, two key questions remain unanswered about all these designs. First, how effective is staging at restoring the performance of bulk transfers? Second, how much storage does staging need? We answer these questions below.

3.5.3. Evaluation

To understand the performance benefits from staging transfers and to estimate the storage staged transfers would need, we implemented and analyzed a simple proxy-based staging service design in the ns-2 simulator. We evaluate the staging service using the network topology and methodology from Section 3.4. We still focus on the performance of a single bulk flow traversing a pair of traffic shapers. However, we now place a staging proxy on the network path between the two traffic shapers, as shown in Figure 3.15. The proxy breaks the bulk transfer into two independent subtransfers by fetching data traversing the first traffic shaper into a local store before sending it across the second traffic shaper.

Does staging improve performance of end-to-end transfers?

We first analyze simulation results from the idealized scenario when there is unlimited amount of storage at the staging proxy. The performance of transfers under this scenario represents an upper bound on the potential benefits from staging. Figure 3.16 compares the loss in performance suffered by our long-running flow traversing two traffic shapers with and without the staging service. The performance loss is computed relative to the

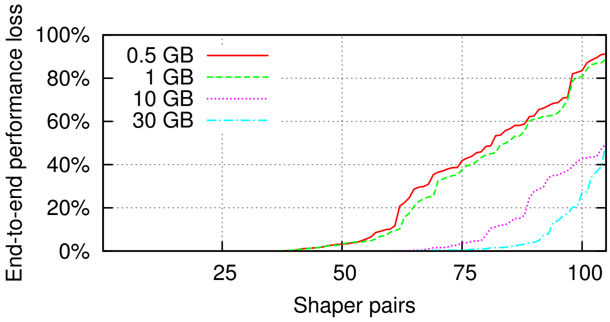


Figure 3.17: Performance loss with the staging service for different maximum amounts of per-flow storage: Reducing the available storage causes the staging service to become less effective.

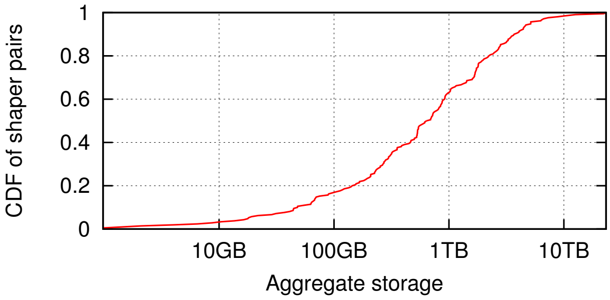


Figure 3.18: Estimate of the aggregate storage required for the staging service: The total amount of storage required very rarely exceeds 10 TB.

minimum performance of the flow when it is traversing either of the two traffic shapers individually and it is computed in terms of the number of bytes the flow transfers during our simulation. The figure shows that staged transfers perform significantly better than end-to-end transfers without staging. In fact, with staging, we see no performance loss when traversing multiple traffic shapers in almost 90% of the cases. This suggests that a staging service could be very effective in counteracting the harmful global slowdown in the performance of bulk flows.

How much storage is needed?

In practice, only a limited amount of storage will be available at a staging proxy. So we repeated the experiments limiting the amount of data our bulk flow can buffer at the staging box. Figure 3.17 shows how our long-running bulk flow's performance varies with different storage limits. As expected, the transfer performance improves when more storage is available. However, the performance benefits are considerable, even when only a small amount of storage is available for staging. When we allocate 30 GB to the bulk flow, the performance approaches the optimal performance we observed with unlimited storage (see Figure 3.16).

Our results suggest that the amount of storage that we would have to allocate for each bulk flow, while not extremely large, is considerable. However, one key question remains: how much aggregate storage does one have to deploy for all bulk flows crossing an access link? It is not possible to answer this question directly from our simulations as we have only one staged bulk flow traversing both traffic shapers simultaneously. However, we can derive an estimate of the aggregate storage required for all flows as follows: we first multiply the storage consumed by our bulk flow by the number of bulk flows that are actively traffic shaped at different times of the day and then compute the maximum storage that would be required at any time during the course of the day. Figure 3.18 plots the results of this estimate. In 50% percent of the cases, the aggregate storage required for staging all bulk flows at university access links is less than 1 TB. In 97% of the cases, the storage required is less than 10 TB. In general, we believe that the performance benefits available from staging justify the required storage.

3.6. Summary and discussion

We conducted a systematic analysis of traffic shaping. Even though traffic shaping is widely deployed today, at the time of writing no known studies investigate the effect of different traffic shaping policies on real network traffic.

We compared different traffic shaping policies inspired by real-world examples. We identified a local traffic shaping policy that greatly reduces peak network traffic (by about 50%) while minimizing the impact on the performance of the shaped flows. However, we also found that multiple of these traffic shapers in the path of a bulk flow can have a significant impact on its performance. To counteract this negative global effect, we propose staging, i.e., breaking end-to-end connections at multiple points in the network. The staging points need storage to temporarily buffer the data of bulk flows in transit. Our evaluation shows that staging is effective at restoring the performance of traffic shaped bulk transfers and can be implemented with a reasonable amount of storage.

Our work explores the implications of the confluence of a number of recent trends. First, ISPs charge their customers based on their peak levels of utilization because meeting their Service Level Agreements requires provisioning their network internally for these peaks. Second, there is significant diurnal variation in bandwidth demand within individual ASs. We found a factor of two variation between 95th percentile utilization and average utilization in our traces. Third, the vast majority of bytes are consumed by large flows. Across all our 35 access traces we found that, on average, more than 70% of the total bytes belong to flows larger than 10 MB. At the same time, on average, less than 1% of the total flows are larger than 10 MB. These large flows are typically not interactive in nature and hence relatively small delays in their completion time are likely to go unnoticed.

These trends taken together have led many ASs to impose traffic shaping on subsets of their traffic to reduce their peak levels of utilization, and to correspondingly reduce their bandwidth costs. Many existing techniques act as blunt instruments, arbitrarily restricting or shutting down entire application classes without considering whether peak levels of utilization will decrease as a result of the traffic shaping. For example, rate limiting peer-to-peer traffic in the middle of the night is unlikely to reduce the 95th percentile in bandwidth consumption.

We showed how ISPs can take advantage of the wide variation between peak and average utilization to effectively smooth out bandwidth peaks. That is, assuming that bulk transfers are relatively insensitive to small delays in their completion time, we explore the use of traffic shaping policies that target large flows during times of peak utilization. Our policies employ multi-level queues to ensure only moderate average delays in the completion time of bulk flows.

Taken together, our proposed traffic shaping policies hold the promise of significantly reducing peak bandwidth utilization, e.g., by a factor of two or more, with no penalty for interactive traffic and minimal average slowdown of bulk transfers. Finally, the traffic shaping mechanisms themselves are inexpensive, suggesting that they can be implemented in hardware and at high speed.

These benefits along with limited drawbacks suggest that an increasing number of ASs may employ these techniques to reduce their bandwidth costs. Perhaps unexpectedly, we find that what appears to be near-optimal local traffic shaping techniques may lead to global harm for bulk flows when widely applied. As more ASs perform traffic shaping, the probability that some AS between a source and destination is performing traffic shaping at a particular point in time increases. Interestingly, the bandwidth available to a bulk transfer decreases with the number of traffic shapers active on its path for two reasons. First, because traffic shaped flows transfer most of their bytes during short time windows, even a small misalignment of these time windows among two shapers results in a large loss in performance for a flow that traverses both. Second, a TCP flow that traverses multiple congested links (such as traffic shaped links) is at serious disadvantage when compared with TCP flows that only traverse one bottleneck. This disadvantage causes an additional performance loss to flows that traverse multiple traffic shapers.

If the trends we observed hold, many bulk transfers will essentially receive no bandwidth. This limitation would come at a time when the demand for transferring large data items across the network is high (consider, for example, high-definition video downloads or large scientific data sets). In this context, we will require alternative architectures to support bulk data transfers. These architectures must consider the economic incentives that led to the traffic shaping in the first place.

One scenario is for ISPs to stop charging for peak levels of utilization and instead adopt a different pricing model. One such model could be to charge on a per-byte basis. Unfortunately, such charging is likely to result in additional imbalances because it does not recognize that "all bytes are not created equal". Not encouraging data to be sent during times of otherwise slack usage means that network resources that must still be provisioned for peak demand sit idle. More importantly, per-byte charging would introduce even larger incentives for ASs to more aggressively traffic shape bulk traffic.

Another alternative, which we explored in our work, is to introduce staging for bulk transfers that traverse multiple traffic shapers. We analyzed two possible designs to achieve staging: one based on HTTP proxies and one on Split-TCP boxes. In the case of HTTP proxies, applications need to be redirected to appropriate HTTP proxies where the bulk transfer is going to be staged. In the case of Split-TCP, a Split-TCP box needs to transparently interpose in the TCP stream and stage the data. In both cases, specialized hosts need to be placed at key locations in the network core. HTTP proxies could be deployed by CDNs like Akamai, which could then offer a staging service. Transparent solutions based on Split-TCP need to be implemented directly by transit ISPs.

4. NetEx: Opportunistic bulk content delivery

In this chapter, we present the design and evaluation of NetEx, a routing and traffic management mechanism that ISPs can deploy in their network to make their excess capacity available to bulk applications. Using real data from Tier-1 ISPs, we quantify how much latency-insensitive bulk content NetEx can deliver using the bandwidth currently lying idle in transit ISPs.

Despite the high cost of transit bandwidth, many backbone links in transit ISPs exhibit low link utilization, when averaged over a day. Studies of different backbones have consistently shown that the average usage of the network links tends to be significantly lower than their capacity. We summarize the results of a few of these studies in Table 4.1. There are two fundamental reasons for the low usage of network links.

1. **Diurnal variation in network load** The traffic of many Internet links is subject to strong diurnal patterns. We show an example of this in Figure 4.1, which plots the traffic load on a backbone link from a Tier-1 ISP over one week. The traffic exhibits clear diurnal patterns, with traffic peaking around 6pm and bottoming in the early morning, resulting in daily peak-to-trough ratio of about 2:1. Similar ratios have also been observed in other backbone networks [BDJT02]. To avoid congestion, backbone ISPs must provision their links for peak utilization. Therefore, during off-peak times, links are sparsely used, thus contributing to the low average utilization of links in the long-term.

2. **Overprovisioning links to satisfy SLAs** In addition to provisioning for peak load, many backbone ISPs intentionally overprovision their links beyond the ex-

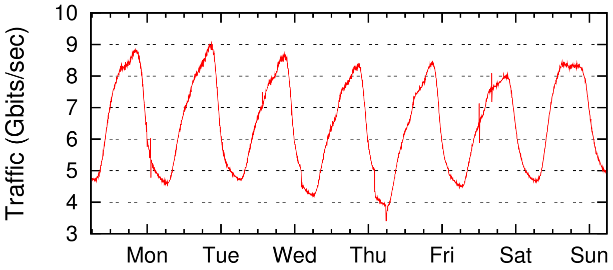


Figure 4.1: Traffic load on a Tier-1 backbone link: Traffic exhibits very clear diurnal patterns during every day of the week.

Study	**Observations on diurnal average load**
Sprint backbone [NTBT04, Spra, IBTD03]	load $<$ 50% for most links load $<$ 30% in 70% of links
Abilene research backbone [ABI]	most used link has load $\simeq$ 18%
Anonymous Tier1 backbone	most used link has load $\simeq$ 60%

Table 4.1: Observations from studies of backbone links: Backbone links exhibit low levels of utilization when the load is averaged at diurnal time scales.

pected peak load to satisfy their service level agreements (SLAs). These SLAs typically specify guarantees on packet delays, jitter, and loss rate. For example, Sprint backbone guarantees a maximum average packet delay of 55 ms and a maximum average packet loss rate of 0.3% between its North American customers [Sprb]. Performance guarantees at packet-level are important for interactive applications, like VoIP and web downloads, which perform poorly when packets are lost or delayed. To avoid lost or delayed packets, network operators often overprovision their links with a capacity considerably higher than the peak traffic load.

Further, SLAs require backbone ISPs to achieve quick recovery from link failures. Most pairs of backbone routers have at least two disjoint paths between them [TMSV03]. If network links are lightly loaded due to overprovisioning, ISPs can quickly and transparently recover from a failure by rerouting traffic to another path [IBTD03].

3. **Overprovisioning when upgrading links** Significant unused link capacity can result when backbone links are upgraded to a higher capacity. Upgrading backbone links is usually a very involved process. To avoid frequent upgrades, ISPs have to provision links for expected growth in traffic over few to several years into the future. Further, standardization of link technologies force upgrades to increase bandwidth in large chunks, even if it leads to unused capacity. For example, an OC-12 (622 Mbps) link is typically upgraded to an OC-48 (2.5 Gbps) or OC-192 (10 Gbps) but nothing in between.

4.1. Related work

4.1.1. Current traffic engineering practices

While transit ISPs do not usually differentiate traffic, they do employ traffic engineering to minimize latency and the likelihood of congestion in their networks. Minimizing latency is important for enabling QoS-sensitive services like VPNs, and for increasing the performance of interactive applications like VoIP or web-browsing. Reducing the likelihood of congestion is important because sudden traffic surges can potentially congest even overprovisioned links, thus causing losses and increased latency. In general, minimizing congestion means distributing the traffic among more links or routing around hot spots, and can thus conflict with minimizing latency which demands that traffic be sent along the shortest path in the network.

Several automatic techniques have been researched that dynamically compute network paths that reduce the likelihood of congestion while limiting latency. These techniques can involve multi-commodity flow problems [GK06, FT00], predictions on future network load [BU97], or a combination of both approaches [SWB$^+$03].

Minimizing congestion and latency are conflicting goals. Existing traffic engineering techniques generally try to strike a balance between these goals [GK06]. This is because transit ISPs do not differentiate traffic, and thus traffic engineering has to operate on the whole network traffic without knowing which application generated it. To understand why this makes a difference, consider two very different applications, BitTorrent and VoIP. When computing routes for BitTorrent, high-bandwidth routes should be preferred, because BitTorrent is a bulk data application that consumes a lot of bandwidth. On the other hand, when dealing with a latency-sensitive application like VoIP, minimizing latency by picking short routes is clearly more desirable. The problem is that, if nothing is know about the application, traffic engineering has no choice but to find a compromise between latency and congestion reduction. On the other hand, NetEx differentiates bulk from normal traffic and routes bulk traffic taking only bandwidth into account and completely ignoring latency.

4.1.2. Using spare capacity in Internet backbones

There are a few examples of past work that tried to exploit spare capacity in Internet backbones. The piece of work that most closely resembles NetEx is perhaps Shaikh and Rexford's [SRS99], who proposed a routing schemes that allocates a fraction of each link's bandwidth to long-lived flows. Long-lived flows are associated with a bandwidth requirement and routed along paths that can provide the needed bandwidth. There are three important differences between NetEx and Shaikh and Rexford's work. First, their approach doesn't differentiate traffic when scheduling packets over network links. This means that packets from long-lived flows and normal traffic are sent with the same priority. Therefore, long-lived flows may congest the links and affect the performance of normal traffic. This cannot happen with NetEx because normal traffic is strictly isolated from bulk. Second, there is no globally coordinated routing. On the contrary, ingress routers select the path for each new long-lived flow without taking into account traffic entering at other routers in the network, and giving preference to shortest paths. One of our contributions is showing that both global coordination of bulk flows and traffic differentiation are necessary to fully utilize the spare capacity present in the network. Third, Shaikh and Redford assume that each long-lived flow is associated with information about its bandwidth requirements, which is used to select the flow's path. Furthermore, whenever a new long-lived flow is identified, the information is signaled along the selected path in order to reserve the bandwidth needed by the flow. In contrast, NetEx does not assume information on bandwidth requirements, but instead uses past network traffic history to estimate future traffic demands. Also, NetEx does not require any global operation, such as signaling, whenever a new bulk flow is identified.

Another early attempt to use spare capacity to send low priority traffic was the QBone Scavenger Service [ST01a]. Scavenger performed traffic differentiation but did not route traffic along routes with the most available spare capacity, as NetEx does.

4.2. Design of NetEx

4.2.1. Overview

NetEx design was guided by two goals. First, NetEx should allow transit ISPs to make efficient use of spare bandwidth in their backbone links. Second, the design should be easy to deploy in practice and must offer immediate, even if incremental, benefits to ISPs that deploy NetEx. At a high-level, NetEx design has two primary components: *traffic differentiation* and *bandwidth-aware routing*. The first allows bulk transfers to exploit spare bandwidth without interfering with existing traffic, while the second achieves efficient use of the spare resources.

1. **Traffic differentiation** Traffic differentiation is necessary to allow bulk transfers to use left-over bandwidth without penalizing normal Internet traffic. NetEx separates traffic into two basic classes: normal traffic that is delay sensitive and bulk traffic that is delay tolerant. Normal traffic is forwarded without any change, while bulk traffic is forwarded in the background with lower priority (i.e., bulk traffic is sent only when there is no foreground traffic waiting to be sent).

2. **Bandwidth-aware routing** To address the limitations of traditional Internet routing in using excess capacity efficiently, NetEx employs bandwidth-aware routing that is optimized to make the best use of available spare capacity. NetEx uses the recent state of the network links to estimate the spare link capacities and bulk traffic load in the near future [LPC$^+$04, LSRS09]. NetEx uses these short-term estimates to cast the routing problem as a standard maximum concurrent flow problem [SM90] that can be solved using a linear solver [SM90]. The resulting routes can efficiently exploit spare bandwidth across potentially multiple paths between a source and a destination. NetEx also periodically recomputes its routes to account for changes in network conditions. We show later in this section that bandwidth-aware routing could be deployed by ISPs today with little to no cooperation from end hosts.

In the rest of this section, we first describe in detail how NetEx can be deployed within a single transit ISP. Later we discuss how NetEx can be incrementally deployed across different ISPs in the Internet.

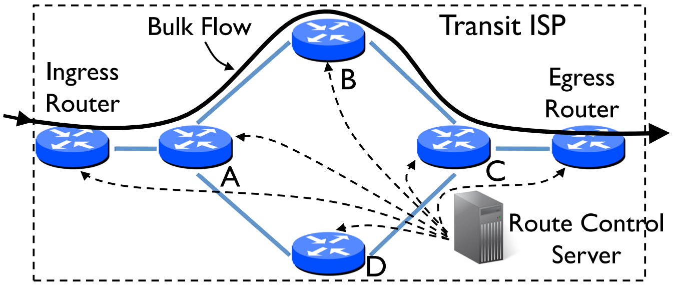


Figure 4.2: Deployment of NetEx in a transit ISP: Bulk transfers are routed through the ISP following the routes computed by the Route Control Server (RCS).

4.2.2. Deploying NetEx within a single ISP

Figure 4.2 illustrates how a transit ISP could deploy NetEx within its backbone. At a high-level, NetEx identifies and classifies bulk traffic at the border routers of the ISP. The bulk traffic so identified is then routed according to routing tables computed and disseminated by a central Route Control Server (RCS). RCS computes the routes using NetEx's bandwidth-aware routing algorithm. When forwarding packets along the routes, NetEx routers forward bulk traffic at a strictly lower priority than normal Internet traffic. In the rest of the section, we describe each of these components in greater detail.

Traffic classification and path selection by border routers

NetEx requires border routers to classify packets as belonging to bulk or normal traffic as soon as they enter the ISP. Border routers mark bulk traffic by setting the DSCP field in the IP headers of the packets [rfc98]. Border routers could automatically classify traffic using deep packet inspection (DPI) or based on flow sizes or durations. Many ISPs are known to use such traffic analysis techniques to identify and traffic shape bulk data traffic today [paca, DMHG08], without any assistance from end hosts. Alternately, end hosts could collaborate with their ISPs by explicitly marking their bulk traffic.

After it has identified a bulk data packet, the border router selects an MPLS path [RVC01] for the packet. The path selection process requires looking up two routing tables. First, the ingress border router (i.e., the router where traffic enters the ISP) looks up the BGP routing table to determine the egress border router (i.e., the router through which traffic must exit the ISP).

Second, after determining the egress router, the ingress router looks up the NetEx routing table, which is computed by the RCS route control server. This table contains the currently active MPLS paths that connect the ingress router to the egress router. Each path is associated with a weight proportional to the fraction of bulk traffic between the ingress and egress routers that the path has to carry. Because it splits traffic among paths in proportion to these weights, the ingress router balances the load according to the strategy that best uses the spare capacity of links.

We now describe these two operations in greater detail.

Egress router selection To discover the egress router, ingress routers keep a table T_1 mapping each network prefix np to the IP address $exit$ of an egress routers. Each entry in the table is a pair $[np, exit]$. The address lookup works as follows. The router performs a longest prefix match of the IP destination address of the packet against the prefixes in the table. The longest matching prefix wins, and the corresponding egress router is selected. The information contained in T_1 is similar to the content of the BGP Loc-RIB. Thus, T_1 can be filled directly by BGP or be even replaced by the Loc-RIB to reduce the memory footprint.

MPLS path selection Finally, the ingress router indexes another table T_2 to discover the set of existing MPLS paths to the destination egress router. Each entry of the table has the form $[exit, \{(l_1, w_1), \ldots, (l_n, w_n)\}]$, and associates an egress router with a list of MPLS paths. Each path is identified by an MPLS label l_i and annotated with a weight w_i. The weight is proportional to the fraction of the total bulk traffic between the ingress and the egress router that each path has to carry. Both the labels and the weights are computed centrally by the RCS. Paths and weights are directly derived from the solution of a multi commodity flow problem, and correspond to the routing strategy that best uses the spare capacity of links. Because packet reordering can negatively impact the performance of TCP flows, it is important to route all packets belonging to a flow along the same path. To ensure this and at the same time avoid keeping per-flow state at routers, we use the packet-header based hashing technique described in [HR08]. This technique selects the fields in packet headers that uniquely identify a TCP flow (i.e. source and destination addresses and ports), computes a hash of these fields, and assigns packets with the same hash value to the same path. To achieve this, the hash space is partitioned according to the number of paths and their weights. That is, each path is associated with a part of the hash space whose size is proportional to the path's weight. In this way, the number of flows routed along a path will be proportional to the path's weight. This hashing technique is simple and does not require any per-flow state in routers, thus reducing their memory requirements. However, since flows vary in size and bitrate, hashing does not guarantee accurate load balancing when the load is measured in bytes. If very accurate load balancing is required, the more complex *flowlet cache* [HR08] could be used.

Route computation and dissemination by Route Control Server

In NetEx, routing of bulk transfers within an ISP is globally coordinated by the Route Control Server (RCS). To adjust to changes in available bandwidth, the RCS periodically recomputes the routing strategy using NetEx's bandwidth-aware routing algorithm. During each recomputation, the RCS uses link utilization and traffic demands observed during the previous time interval as inputs. The RCS periodically collects this information directly from the routers, for example using SNMP [CFSD90]. Once the new routing strategy is computed, the RCS disseminates the routing information to all routers to make sure that bulk traffic is routed according to the new strategy during the next time interval.

Bandwidth-aware routing algorithm: The goal of bandwidth-aware routing is to maximize the amount of bulk data that can be delivered using the spare bandwidth on network links. We cast the routing problem as a maximum concurrent flow problem [SM90] for which there are efficient solving techniques. The inputs to the optimization problem are (a) the network topology, (b) the available link bandwidths, and (c) a matrix containing the traffic demands for each flow in the network. Each flow is identified by the corresponding ingress and egress routers.

We tested two problem formulations that optimize for different objectives. The first is a maximum concurrent flow problem [SM90] whose goal is to maximize the total load delivered by the network, while preserving the ratios among the elements in the demand matrix. The second is a formulation previously proposed for routing traffic within a Tier-1 ISP [FT00] with the goal of balancing load across multiple paths and lowering the peak utilization on links. We compare the performance of the two formulations experimentally in Section 4.3.

To solve both optimization problems we use CPLEX [cpl], a commercial optimization software. The output of the algorithm specifies, for each router in the network, the fraction of a given source-destination flow that should be routed on a given outgoing link. This information is used to generate the routing paths for the entire network.

Because the propagation of routing information to all routers is not globally synchronized, there may be periods of time when the information at different routers is inconsistent. To mitigate this problem, the RCS always generates unique and monotonically increasing MPLS labels for the new paths it computes. When they receive the new labels, the routers discard the old ones. If a router receives a packet tagged with an old MPLS label that it doesn't recognize, it simply drops the packet. Thus, routing inconsistencies during the short period of time when routes are updated would lead to dropped packets. Our evaluation in Section 4.3 shows that routes only need to updated once every 30 minutes, thus suggesting that the performance loss due to routing updates will be small. We confirm this intuition through an experiment in Section 4.2.4.

Packet forwarding by NetEx routers

NetEx establishes MPLS tunnels between ingress and egress routers along the paths determined by the RCS. To identify the path along which a packet must be routed, the ingress router prepends an MPLS header to the IP packet. The MPLS header contains a label identifying a path, and each router along the path keeps a forwarding table that maps the label to the next downstream router. MPLS packets are forwarded hop-by-hop using these tables until they reach the egress router, where the IP packet is decapsulated and routed to the adjacent ISP.

When forwarding packets, NetEx routers send bulk traffic with strictly lower priority than normal traffic. NetEx routers use two queues, one for normal traffic and the other for bulk traffic. Packets from the bulk traffic queue are sent only when there are no packets in the queue for the normal traffic. By giving strict priority to existing traffic, NetEx ensures that bulk traffic does not affect normal traffic and that bulk traffic is limited to only using the spare bandwidth. Many Internet routers already support

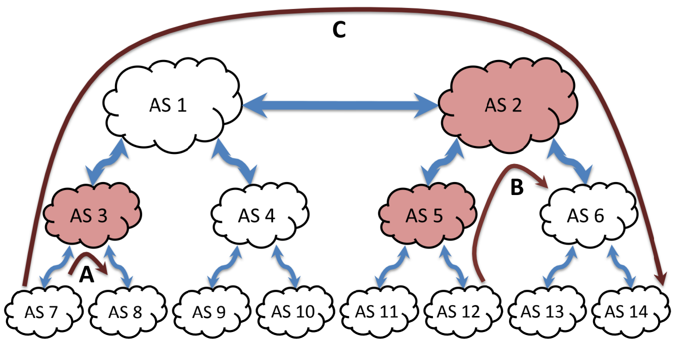


Figure 4.3: Partial deployment in an inter-domain setting: NetEx can be partially and independently deployed by ASes. Shadowed ASes have deployed NetEx, and each vertical link represents a customer-provider relationship.

multi-queue scheduling [Cisa] and ISPs can prioritize traffic by simply enabling such scheduling.

4.2.3. Deploying NetEx across multiple ISPs

NetEx's bandwidth-aware routing does not require any changes to inter-domain routing protocols. Inter-domain bulk transfers are routed along the same inter-AS paths computed by BGP today. Although there may be additional benefits to be gained by having a cross-ISP implementation of NetEx, this would require major changes to BGP, a complex protocol with many specialized policies and mechanisms. Moreover, to deploy NetEx across multiple networks, ISPs would have to disclose information about their topologies and traffic matrices, something that ISPs have been reluctant to do in the past. We therefore suggest that ISPs independently and incrementally deploy NetEx, and route inter-domain bulk transfers on standard BGP paths.

Figure 4.3 illustrates a partial deployment scenario where a fraction of the ISPs have independently deployed NetEx. We discuss the benefits of this partial deployment for three bulk transfers labeled as A, B, and C in the Figure. Bulk transfer A occurs between two customers of an AS that deploys NetEx. In this case, both customer ASes 7 and 8 benefit from the fact that their transit provider, AS 3, deploys NetEx and can deliver the data cheaply and efficiently end-to-end. Next we consider a bulk transfer (labeled as B in the figure) traversing multiple transit ASes (ASes 5 and 2), all of which deploy NetEx. In this case, ASes 12, 5, and 6 also benefit from the lower rate charged by the respective providers. The last bulk transfer (labeled as C) traverses some ASes that deploy NetEx, and some other ASes that do not. In this case, only customers of ASes that deploy NetEx benefit. ASes that do not deploy NetEx simply route the transfers using their usual intra-domain routing protocols. This ensures that NetEx can be incrementally and independently deployed by ISPs, and that NetEx provides incremental benefits to each adopting ISP and its customers.

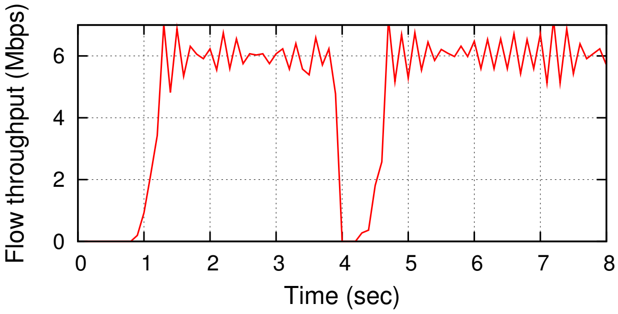


Figure 4.4: Effect of path rerouting on the throughput of bulk flows: A bulk flow takes approximately one second to fully recover from a change in the routing path.

4.2.4. Prototype implementation

To ensure that we have not ignored any detail or subtle issues in our design, we implemented a fully functional prototype of NetEx and verified that all the forwarding and routing techniques work.

Our NetEx prototype was implemented on Linux. The prototype consists of a software router and a *router control server* (RCS) process. We used the Click [KMC+00] modular router running in kernel mode to implement the router; it is based on the Click reference specification of a standard IPv4 router [KMC+00] which we extend with five new Click elements to provide the additional NetEx functionality. In addition, a colocated daemon process running in user mode collects traffic information, sends this information to the RCS and updates the router's tables using the information it receives from the RCS. The RCS is implemented as a user level process that we envision being deployed on a dedicated server within the ISP. The routers and the RCS communicate using a TCP connection. The Click elements for our router are implemented in C++ (2347 lines of code), while the daemon and the RCS are implemented in Perl (2403 lines of code).

Prototype evaluation

We deployed the prototype on Emulab to study how well various routing and forwarding techniques work. Here we present an example result that shows the effect of periodic routing changes on the performance of bulk TCP flows routed through NetEx. We emulated the network topology of Figure 4.2 using the Emulab [EMUb] testbed. Specifically, we used machines with 850 MHz Intel Pentium III processors, 512 MB of physical memory and 5 10/100 Mbps Ethernet ports. Links have 10 Mpbs capacity and 10 ms delay. The NetEx router is deployed on all machines, and the RCS is deployed on the ingress router.

We initiated a single TCP transfer from the ingress to the egress router. The initial path traverses nodes A, B and C. After five seconds, the flow is rerouted through node D. Figure 4.4 shows the evolution of the flow's throughput during the first ten seconds.

As we can see, changing the routing path causes a sharp drop in throughput, from which the flow recovers completely after approximately one second. Given that NetEx routing paths need to be recomputed only once every 30 minutes (see Section 4.3), such a sporadic drop in throughput is negligible to the performance of long-lived bulk flows.

4.3. Evaluation of NetEx

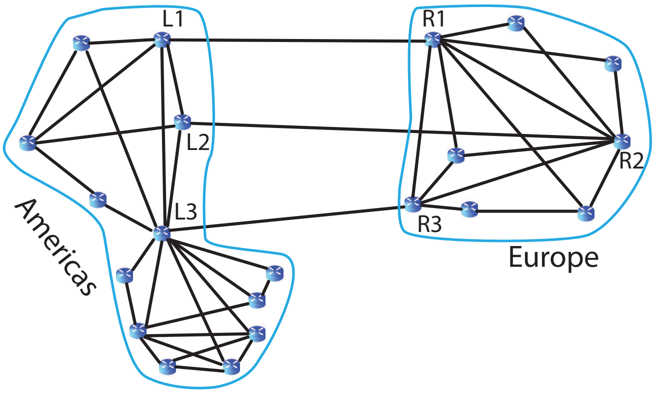
In this section, we study how well NetEx would perform when deployed within a single large transit ISP. In particular, we are interested in answering three questions: (a) How much more bulk data can NetEx send compared to current shortest path routing? (b) How well do bulk flows perform, i.e., what throughput do they get? and (c) Which aspects of the system and the input (topology/traffic matrix) are more important for shaping the final gains?

4.3.1. Methodology

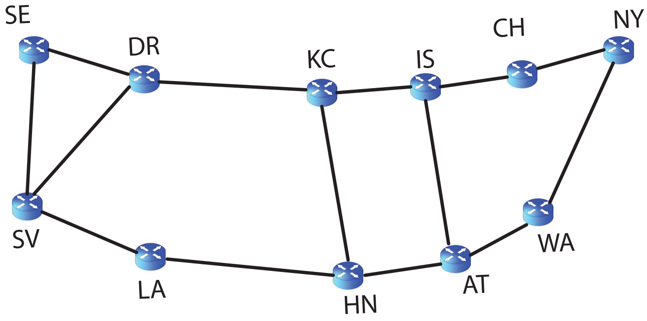
It is hard to scale our prototype implementation deployed on Emulab to the number of routers and links in a large ISP network. So we reimplemented NetEx in the ns-2 network simulator and used it for our evaluation.

However, simulating high-speed multi-gigabit backbone links is computationally very expensive even in ns-2. Therefore, we use a well-known technique [PPPW05] to scale down the network capacity, link usage, and the traffic matrices by a factor of at least 1000, depending on the simulated topology [1]. While scaling down the traffic matrices and link usage, we preserve their relative proportions as well as the observed diurnal patterns, thus allowing the results to be scaled back.

Ideally we would have liked to run actual TCP flows for both normal and bulk traffic. However, our machines ran out of memory simulating hundreds of thousands of TCP flows that run simultaneously in large ISPs. Therefore, in order to make the simulation tractable, we did the following. First, we don't simulate the large number of individual TCP flows that constitute the normal traffic; instead, we traffic shape the bulk TCP flows so that they only use the spare bandwidth left unused by the normal traffic on each link. Second, we don't simulate bulk TCP flows smaller than a certain size. Instead, we partition bulk traffic into TCP flows of fixed, large size: 4GB for Abilene and 40GB for the Tier-1 ISP due to higher scaling factor used. In order to quantify the performance of flows smaller than 4GB, we assume that each of the large 4GB flows comprises several smaller "virtual" subflows. After the simulation, we divide the bandwidth consumed by each large flow equally among the smaller virtual subflows. This requires assuming that bulk TCP flows that are larger than a few MBs but smaller than 4GB would divide bandwidth equally between themselves. We believe that this is a reasonable assumption given our difficulties simulating such small flows.



(a) Tier-1 ISP



(b) Abilene

Figure 4.5: Network topologies used to evaluate NetEx: A large commercial Tier-1 ISP with 21 PoPs and 43 bidirectional links (a) and the Abilene research backbone with 11 PoPs and 14 bidirectional links (b).

Data from real-world ISPs

The input to our simulations are the network topology and traffic matrix of an ISP. We use data from two large backbone networks: the Abilene research backbone [ABI] and a large commercial Tier-1 AS offering transit service to access ISPs in multiple continents.

1. **Topologies**

   Figures 4.5 (a) and (b) show the topologies of the Tier-1 and Abilene ISP backbones respectively. The Tier-1 network offers transit services to more than 40 access ISPs most in Europe and in the Americas, where it also peers with 200 other Tier-1/Tier-2 networks and major distributors of content. The backbone PoPs in the Tier-1 ISP are interconnected using one or multiple 10 or 40 Gbps links. All links in Abilene are OC-192 optical fibers with 10 Gbps of capacity.

[1] The factor is 1000 for Abilene, and 10000 for the Tier-1 network because of its higher capacity.

2. **Traffic matrices**

   To derive the traffic matrices for Abilene, we used its NetFlow data to compute the intra-domain traffic entering and leaving at each Abilene router. We applied the simple gravity model [GJT04] to estimate the traffic matrix. Our data comprises all traffic sent in Abilene during the week starting on January 8^{th}, 2007. For the Tier-1 ISP, we also obtained 5-minute load aggregates for the traffic entering and exiting the backbone at each one of its PoPs. Our measurements reflect real loads during February 2009. As before we computed the network's traffic matrix using a gravity model.

Simulating foreground interactive traffic

We used ns-2 to implement the topology, the routing, and the interactive traffic as described before. For the Tier-1 AS we used our traffic aggregates to model the interactive traffic load. As Abilene is a research network, its real usage levels are very low (around 3%). It is obvious that in such over-provisioned research network there exists plenty of spare capacity that can be used for bulk data transfers. To make our evaluation more realistic, we choose to scale up the load on Abilene links to what one would find in Tier-1 ISPs.

Simulating background bulk traffic

We attached to each PoP an additional source generating bulk traffic according to the traffic matrix. The bulk sources are connected with links of infinite capacity, i.e., they can use all the available bandwidth given to them. We produce the trace describing the arrival times for bulk data transfers by simulating a Poisson process with rates varying over time according to the diurnal traffic patterns. We generated flows of different sizes according to a distribution observed in the Abilene's backbone.

NetEx Routing

We evaluated several routing algorithms in NetEx. They fall into three broad categories of routing traffic.

1. **Static routing**

   We simulated static least-weight path routing with different weights for the links: geographical distance (DS), simple hop-count (HC), and the real routing weights used in the studied topologies (WE).

2. **Greedy routing**

   We simulated a greedy widest path algorithm [SRS99], where each data source selects the path with the most available spare capacity, independent of other sources. The performance of greedy routing reflects the performance of routing schemes where routes are selected without coordination between different flows. This includes the approach used in [SRS99] and systems that use overlay routing.

3. **Traffic Engineering**

   In stark contrast to greedy routing, traffic engineering computes routes taking the global demand of the network into account. NetEx's bandwidth-aware routing described in Section 4.2.2 falls into this category. As described in Section 4.2.2, we simulated bandwidth-aware routing using two different traffic engineering objectives: one that optimizes for delivering most bulk load [SM90] and one that minimizes the peak load across different links [FT00]. In both cases, we solve the corresponding optimization problems using CPLEX [cpl].

Simulated bulk workloads

We evaluated the performance of NetEx for two different bulk workload models. Each workload model defines its own traffic matrix (i.e., demands between any pair of PoPs in the topology) for bulk traffic. For each traffic matrix, we compute the maximum amount of bulk data that NetEx can deliver while preserving the ratios between the elements of the traffic matrix. To find the maximum, we run a number of simulations, all configured with the same parameters except for a multiplicative factor used to scale the values of the bulk traffic matrix. We run a binary search to find the scaling factor that corresponds to the maximum deliverable workload. At every step of the binary search, a simulation is executed to see whether NetEx can deliver the bulk demand when the traffic matrix is scaled with that factor. If NetEx is able to deliver at least 99% of the bulk demand, the simulation is deemed successful. After a sufficient precision is reached, the binary search ends, and the result of the last successful simulation gives the maximum amount of data that NetEx can deliver under that traffic matrix.

1. **Native workloads**

   The first workload model directly uses the traffic matrices of the real traffic demands as measured in the ISP. We believe that this traffic model is not only the most realistic but also the most challenging. Of course NetEx can also be used to push additional traffic that falls outside the native traffic matrix. In this case it is easy to see that a system like NetEx can drive the utilization of all links to 100% by sending additional bulk traffic on each link with some free capacity. However, such a hypothetical bulk traffic matrix cannot be justified based on data from real traffic matrices. Unless stated otherwise, all simulations result in this section refers to this workload model.

2. **Datacenter workloads**

   The second workload model is motivated by emerging Web 2.0 and cloud computing applications whose data is often hosted in datacenters located in distant geographical areas. To ensure durability and fault-tolerance, these applications often need to replicate data across multiple datacenters. The ensuing data transfers are bulk in nature, can run in the background and are therefore well-suited to NetEx. To evaluate the effectiveness of NetEx in delivering this traffic, we colocated a virtual datacenter with 5 of the 8 PoPs in the European subtopology of the

Tier-1 ISP (Figure 4.5 (a)). We chose the European topology because it's where the Tier-1 ISP has better presence, and selected the 5 best connected PoPs in that topology. The traffic matrices for this model are generated by selecting sender and receiver datacenters and having the senders send as much data as possible to each receiver.

Our evaluation uses two traffic matrices: (a) where only one data center sends data to each of the remaining ones (single source) , and (b) where all data centers send data to all other datacenters concurrently (full mesh). In both cases, we assume that each datacenter has the same amount of data to send and we measure the maximum amount of data that can be delivered on each day. To efficiently distribute the load, we assume that all datacenters form a swarm and cooperate in distributing the data, with data sources acting as seeders.

4.3.2. Overhead of NetEx

Before quantifying the amount of bulk data that NetEx can deliver, we quantify the overhead of NetEx's bandwidth-aware routing algorithm. The overhead is broken down in three components:

1. **Route computation cost at the RCS**

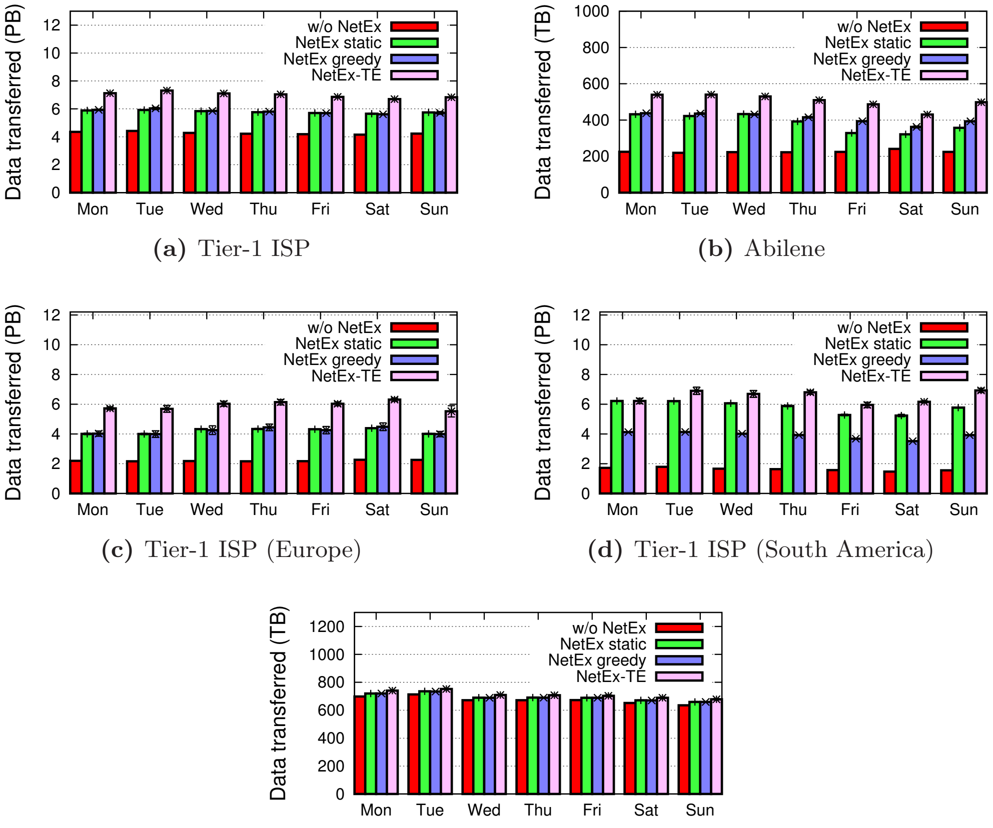
   During our simulations, we computed the routes using the CPLEX [cpl] linear solver on a 2.5 Ghz AMD processor running Linux. The computation of routes for the larger Tier-1 topology took on average 0.1 seconds, and never more than 1.14 seconds. This shows that route computation on a well-provisioned RCS should scale well even to larger topologies. For very large topologies where linear solvers may become inefficient, algorithms do exist that approximate a solution efficiently [Kar02], with complexity polylogarithmic in the number of edges and nodes of the topology.

2. **Bandwidth costs**

   To compute routes, the RCS has to fetch information on load and traffic demands from the routers and distribute the routing information back to the routers. If we encode both link loads and elements of the demand matrix with 16 byte integers, the RCS needs to receive an aggregate of 7.5 Kbytes at every routing update for our large Tier-1 topology. The total routing information produced by the RCS for the Tier-1 topology was never more than 10 Kbytes, and the information shipped to any single router never more than 1.5 Kbytes. Since routes are required to change only every half an hour or more (see Section 4.3.5), these values result in very modest bandwidth requirements.

3. **Additional routing table size**

   In our large Tier-1 topology, the maximum total size of the NetEx routing tables (see Section 4.2.2) and the MPLS lookup tables ever dispatched to a single router was 122 entries, corresponding to 1.5 Kbytes.



(a) Tier-1 ISP

(b) Abilene

(c) Tier-1 ISP (Europe)

(d) Tier-1 ISP (South America)

(e) Tier-1 ISP (North America)

Figure 4.6: Additional data transferred by NetEx: NetEx bandwidth-aware traffic engineering (NetEx-TE) attains the best utilization of spare capacity, increasing the ISP's throughput by up to 340% (d). However, in some topologies with PoPs connected by low-capacity links (e), NetEx does not bring substantial benefits (throughput increases by only 6%).

4.3.3. Bulk data delivered by NetEx

We start by showing the aggregate amount of data, both interactive and bulk, that can be carried by NetEx using the routing algorithms in the three routing categories described in Section 4.3.1. For each of the three routing categories, that is, static and greedy routing, and traffic engineering (TE), we only show results for the routing algorithm that performed best in the category. In particular, in the case of traffic engineering, both formulations described in Section 4.3.1 achieve the same performance. The reason for this will be explained later in Section 4.3.5. Figures 4.6 shows the aggregate bulk data delivered by NetEx during each day of the week in the entire Tier-1 topology, Abilene, and the European, South American, and North American subnetworks of the Tier-1 topology.

Additional bulk data transfer capacity In the entire Tier-1 ISP, the Abilene network and the European subnetwork of the Tier-1 ISP, NetEx with traffic engineering (NetEx-TE) can transfer 60 - 180% more data than what is being delivered today without NetEx. The absolute amount of extra bulk data that can be delivered is almost 3 PBytes per day in the case of the Tier-1 AS. To put things in perspective, such a volume is almost 100 times greater than the raw amount of data produced each day by the Large Hadron Collider of CERN, one of the biggest individual producers of data in the world (27 Tbytes per day [LSRS09]). Amazingly, all this data can be sent using resources that are lying idle today.

Impact of ISP topologies In general, NetEx-TE performs better within continental backbones: in Abilene and the European and South American subnetworks of the Tier-1 ISP NetEx transferred between 80 - 340% more data than what is being delivered today. An exception is the North American Tier-1 subnetwork, where NetEx could not increase the network throughput by more than 6%. The reason for this result is the low capacity of links connecting a single PoP in the North American subnetwork to the rest of the topology. NetEx performs better within most continental backbones (e.g., Abilene and European sub-graph of Tier-1 ISP) because PoPs within continental backbones are usually more densely connected, offering more alternate routes that traffic engineering can exploit.

Bandwidth-aware traffic engineering versus static routing The difference between the performance of NetEx-TE and static routing shows the extent to which traditional inter-domain routing algorithms limit the efficient use of spare resources. In Abilene, NetEx-TE delivers at least 20% more bulk data than static routing. In the Tier-1 ISP, NetEx-TE delivers at least 1 PByte of additional bulk data every day.

Traffic engineering versus greedy routing The difference between NetEx-TE and NetEx-greedy indicates the advantages of coordinated traffic engineering over greedy routing of single flows without any coordination with other flows. In NetEx-greedy, each flow is routed along the path with the most available spare capacity, without taking into consideration other concurrent flows. In the Tier-1 and Abilene topologies, NetEx-TE delivers 20% more data than NetEx-greedy. This result hints at the potential limitations of overlay routing solutions used in P2P systems, where each flow tries to find paths that optimize its throughput independently of other flows. On the contrary, having a global view of the network and coordinating different flows, like NetEx-TE does, brings substantial benefits.

Comparing different traffic engineering objectives As said before, we evaluated two different traffic engineering objectives, one focusing on delivering most data and other on balancing load across the different links. We found that both objectives perform very similarly (not shown in the Figures above) and deliver similar amounts of data.

Size	Example	Streaming rate
15MB	10min YouTube low def	200 Kbps
40MB	10min YouTube med. def	512 Kbps
150MB	10min YouTube high def	2000 Kbps
2GB	iTunes VoD	4900 Kbps
4GB	DVD	9800 Kbps
25GB	Blu-ray disc	61250 Kbps
100GB	Data backup	NA

Table 4.2: Application examples used in our analysis: Each flow size corresponds to a popular bulk application.

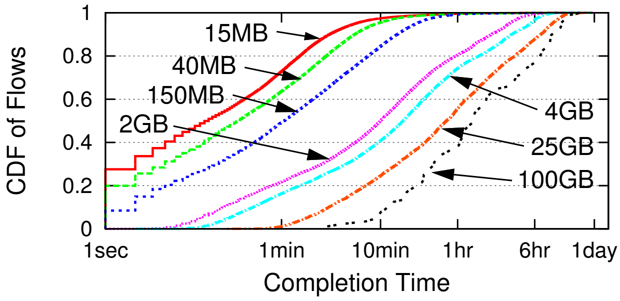


Figure 4.7: Completion time of flows of different sizes routed through NetEx: Most flows achieve good completion times.

As we show later in Section 4.3.5, both schemes achieve near-optimal performance by saturating the links that form a min-cut within the network.

In summary, bandwidth-aware traffic engineering, NetEx-TE, performs considerably better than all other routing schemes. Since the overhead of implementing bandwidth-aware routing is relatively small (see Section 4.3.2), NetEx-TE stands out as the most favorable routing scheme.

4.3.4. Performance of NetEx bulk flows

The previous section showed that NetEx can indeed deliver substantial amounts of additional bulk data. A natural next question is how well the individual NetEx flows perform and whether their performance is acceptable to different types of bulk applications.

To estimate the performance that different applications are likely to experience when their traffic is routed through NetEx, we generated a trace of flows of different sizes, corresponding to a set of popular bulk transfer sizes, as illustrated in Table 4.2. The distribution of flows of different sizes in the generated trace is the same we observed in one of Abilene's backbone links [2]. To estimate the performance these flows would achieve

[2] Although it would have been more compelling to use the flow sizes observed in the Tier-1 network, we unfortunately don't have flow-level information for this network.

when routed through NetEx, we employ the methodology described in Section 4.3.1, in which we divide the bandwidth consumed by our real TCP flows among a number of "virtual" subflows.

Completion times of bulk flows Figure 4.7 shows the time these flows take to complete in the Tier-1 topology when routed through NetEx. For delay-tolerant applications like online backups and DVD downloads, NetEx provides good completion times and performance: a 100GB backup rarely takes longer than 1 day, nearly 80% of 4GB DVD movie downloads take less than 1 hour, and the median download time for a large 25GB Blu-ray disc is 1 hour.

However, for real-time or soft real-time applications, like video on demand, the overall completion time is not the only metric of interest. This is because users start consuming the content (i.e., watching the video) shortly or immediately after the download is initiated. Therefore, in spite of good *average bandwidth* for the download, the content playback may still stop during an occasional period of low bandwidth, hurting users' experience.

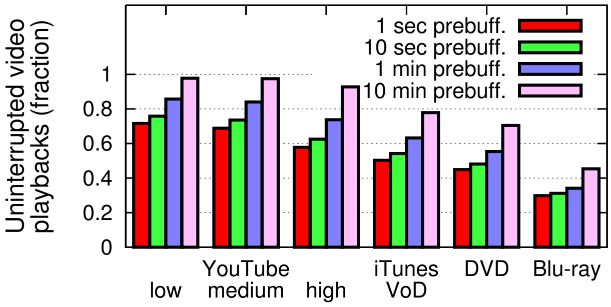


Figure 4.8: Fraction of uninterrupted video playbacks when videos are downloaded through NetEx: The fraction depends on how much prebuffering is allowed before the playback starts.

Suitability for soft real-time apps like video-on-demand To quantify this effect, we simulated a video playback at the appropriate rate (see Table 4.2) alongside our network flows. Figure 4.8 shows the fraction of video playbacks that completed without interruptions, for varying amounts of initial prebuffering time. The prebuffering time refers to the delay between the time when the video starts to download and the time when the user starts to watch (i.e., the playback starts). 70% of low-definition YouTube videos can be watched almost immediately after the download starts. For higher-definition videos, this fraction is smaller. However, 70% of high-definition YouTube videos can still be watched after 1 minute of prebuffering, and 80% of the iTunes movies can be watched after 10 minutes prebuffering. For even larger videos, live DVDs or Blu-ray, prebuffering, even in the order of several minutes, is not as effective.

Our results show that NetEx flows achieve good performance considering that they are routed in the background and use only spare resources. Not only is NetEx well suited for delay-tolerant bulk applications like online backups and DVD downloads, but it also provides surprisingly good performance for video streams with low to medium bitrates. However, although better than expected, the performance is not good enough to warrant the use of NetEx for media streaming applications.

Moreover, NetEx is not suitable for hard real-time or highly interactive applications like video conferencing, online games, or Skype, as it cannot offer any guarantees on minimum bandwidth achieved by flows over short time scales. For example, 30% of the time NetEx is not able to sustain a 200 Kbps data rate (low definition YouTube video) without buffering for several seconds.

4.3.5. How close to optimal is NetEx?

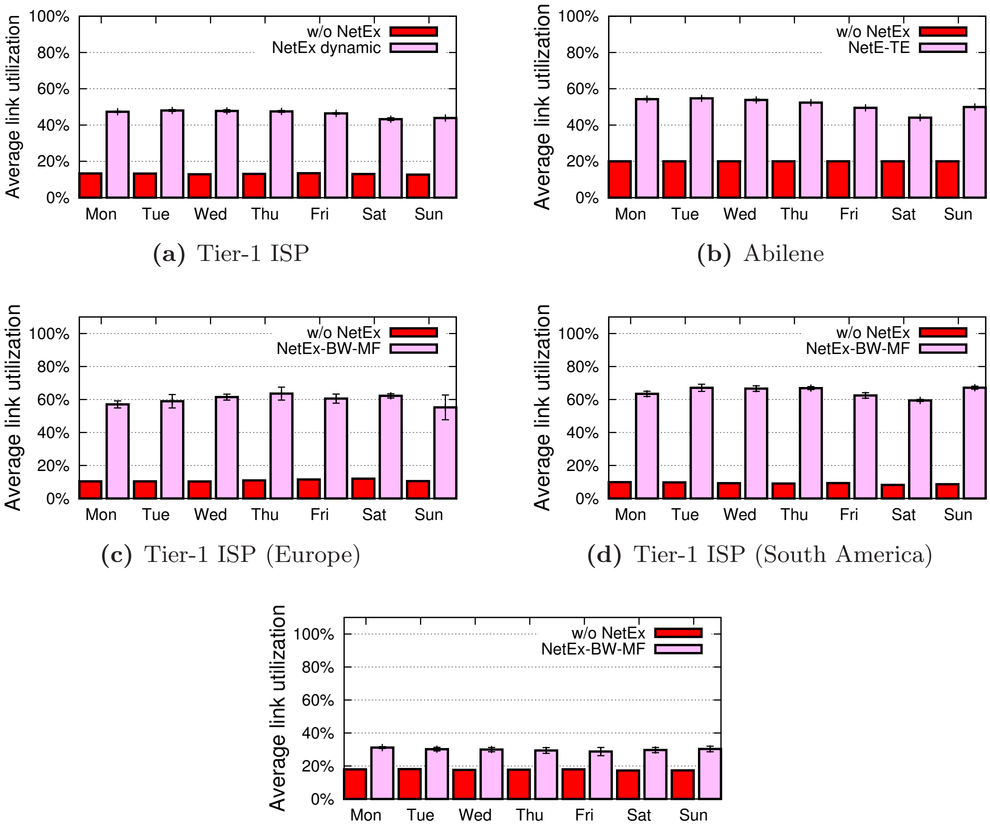
Up to now we have established that NetEx can push substantially more bulk data than what is currently transferred in the network and that the data transfers achieve good performance. In this section we show that NetEx is actually very close to optimal with respect to the amount of bulk data transferred. This means that it is not possible to transfer considerably more bulk data than NetEx does without exceeding the capacity of some link. We will establish this optimality by showing that NetEx almost saturates the capacity of a cut in the network, that is, a set of links that partition the graph.

Figure 4.5 shows the topologies of the Abilene and Tier-1 backbone. For the Tier-1 ISP, NetEx-TE saturates the cut that comprises transatlantic links (L1,R1), (L2,R3), and (L3,R3). When using NetEx-TE, the utilizations of the cut links are, 97%, 99%, and 99%, respectively. For Abilene, the cut has two links, (DR,KC) and (LA,HN), both of which are driven to 99% utilization by NetEx-TE. For the European subpgraph of the Tier-1 ISP, the minimum cut consists of all outgoing links from node L1, whose utilization is 80%[3], 99% and 96%, respectively. Any routing algorithm attempting to deliver more data under the given traffic matrices will be bounded by the capacity of these minimum cuts. This also explains why both our traffic engineering algorithms (described in Section 4.3.1) optimizing for different objectives achieve the same performance. Both saturate the min cut and are ultimately limited by the cut's capacity.

Impact on average link utilization

Saturating a cut in the network does not imply that all other links of the network are fully utilized. In Figure 4.9 we plot for each day of a week the average utilization across all links in our networks. Even though NetEx-TE achieves near-optimal performance, the average utilization across all links in Abilene and the Tier-1 topology is around 50%.

[3] The utilization of link (L1,R1) is only 80% because its capacity (100 Mbps) is 2 orders of magnitude lower than the remaining links (30 Gbps). Therefore, driving this link to 100% utilization does not result in a considerable increase in network capacity.



(a) Tier-1 ISP

(b) Abilene

(c) Tier-1 ISP (Europe)

(d) Tier-1 ISP (South America)

(e) Tier-1 ISP (North America)

Figure 4.9: Average link utilization with NetEx: Results from Tier-1 ISP show that NetEx can increase the utilization of the ISP's links by a factor of 3 or more.

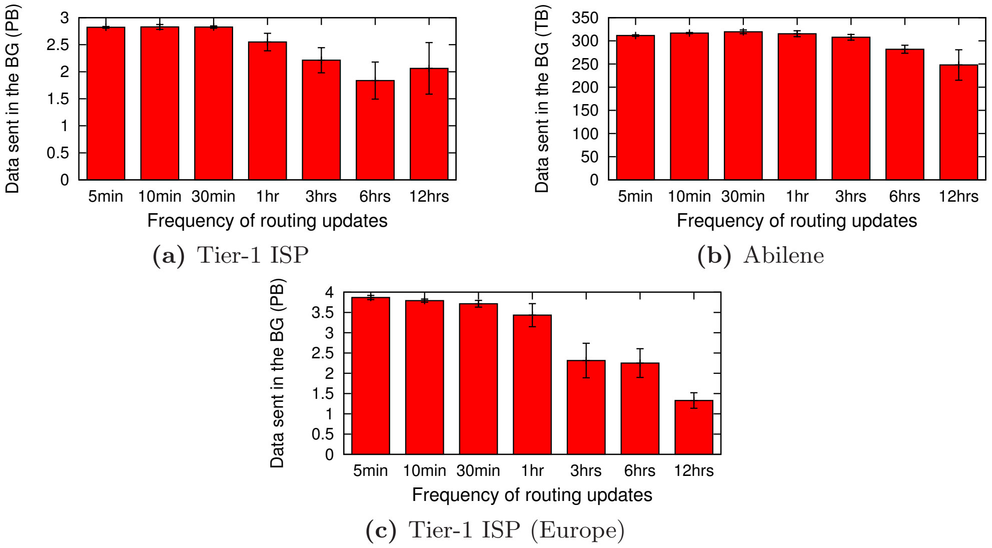
Note, however, that this represents a two to three fold improvement with respect to the link utilization levels without NetEx.

4.3.6. Contributions of different design elements to the performance of NetEx

We now like to quantify the contributions of different design elements to the performance achieved by NetEx. The three design elements are (1) traffic differentiation to delay bulk traffic until spare capacity becomes available, (2) periodic route recomputation to adjust routing to the changes in available bandwidth, and (3) multi-path routing, which enables splitting bulk traffic between multiple paths in the network.

Traffic differentiation

Because traffic differentiation constrains bulk transfers to only using spare capacity, if a NetEx bulk transfer is initiated at a time where the amount of spare capacity is low,



(a) Tier-1 ISP

(b) Abilene

(c) Tier-1 ISP (Europe)

Figure 4.10: Amount of additional background data delivered when routes are recomputed with different frequencies: Results from Tier-1 ISP show that increasing the routing interval to up to 30 minutes does not cause a noticeable decrease in performance.

the transfer is likely to receive little bandwidth and be delayed to a later time when the network is less utilized. To understand how long traffic can be delayed, we plot in Figure 4.11 the completion time for the 40GB transfers initiated in the Tier-1 network during an entire day. When transfers are initiated during periods of high utilization (i.e. around 4pm GMT), they can be delayed by up to 12 hours or more. Interestingly, the median completion time rarely exceeds 2 hours, further showing how NetEx provides good average performance to bulk transfers.

To quantify how much NetEx benefits from delaying traffic and how much from bandwidth-aware routing, it is sufficient to compare the data delivered with static routing (where traffic can only be delayed, but not rerouted) with what is delivered using bandwidth-aware routing in Figure 4.6. In the Tier-1 and Abilene topologies, delaying traffic accounts for slightly more than 50% of the data delivered by NetEx.

Periodic route recomputation

NetEx benefits from recomputing its paths periodically. In Figure 4.10 we plot the amount of additional bulk data transferred by NetEx-TE using different frequencies of route recomputation in the Tier-1 ISP, Abilene, and the European subnetwork of the Tier-1 ISP. In general, NetEx benefits from recomputing paths periodically during the course of the day. In order to achieve the maximum amount of delivered data, recomputations need to be performed at least once every 30 minutes in the Tier-1 ISP. Less frequent recomputations lead to noticeable reductions in the delivered data. In Abilene

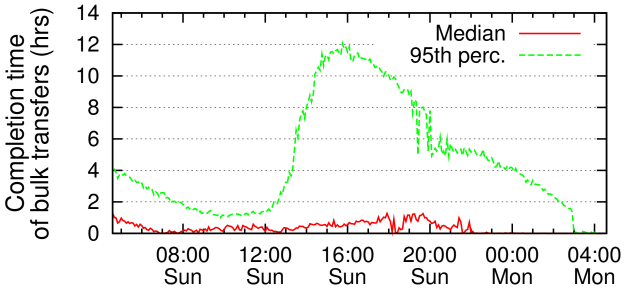


Figure 4.11: Completion time of NetEx bulk transfers started at a given time of day in the Tier-1 ISP: Initiating transfers at times of high network utilization results in longer delays. Each transfer is 40GB.

however, it is sufficient to compute routes every 3 hours; more frequent recomputations do not improve performance. Therefore, the frequency at which the routes must be recomputed varies from one ISP to another.

We believe that the need for more frequent recomputations in the Tier-1 ISP is a result of the different types of traffic carried by the two networks. Because Abilene is a research network connecting universities, most bulk traffic is generated during the central hours of the day when people are working. On the other hand, the commercial Tier-1 ISP carries traffic that is generated more evenly throughout the day with lower peak-to-trough ratio. This calls for more frequent rerouting to account for recently generated traffic.

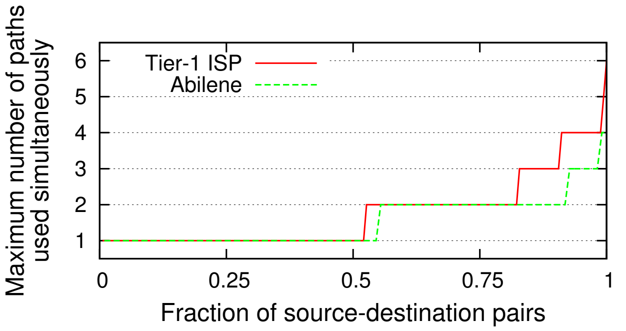


Figure 4.12: Maximum number of distinct paths simultaneously used by NetEx between each source-destination pair: At any time, no more than 6 active paths are used between any pair of PoPs.

Multi-path routing

NetEx bandwidth-aware routing uses spare capacity along multiple paths between a source and a destination to deliver bulk data. In Figure 4.12 we plot the maximum

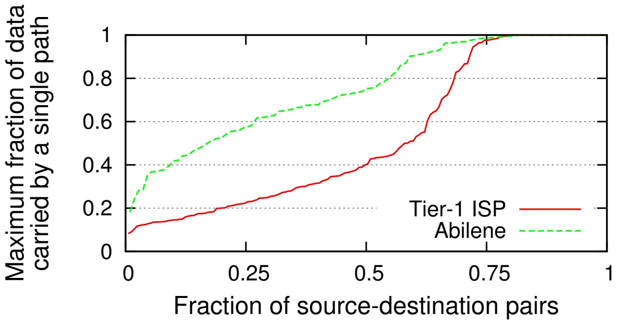


Figure 4.13: Maximum amount of daily load sent on a single path: In most cases, data is split among different paths, suggesting that it is necessary to use them.

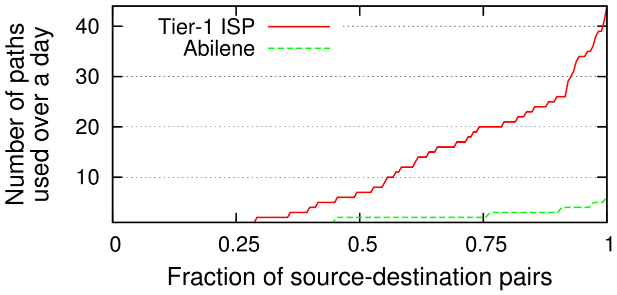


Figure 4.14: Number of distinct paths used by NetEx between each source-destination pair over an entire day: Some source-destination pairs in the Tier-1 ISP were using more than 40 paths.

number of distinct paths that are simultaneously used between any pair of PoPs in our topologies. Almost half of the source-destination pairs simultaneously use two or more distinct paths, while a non-negligible fraction of pairs (20% for the Tier-1 ISP, and 10% for Abilene) use three or more distinct paths at the same time. Thus, relatively few distinct paths are used concurrently for a source-destination pair. However, are these always the same paths or do they change throughout the day? Moreover, how is the total traffic distributed among these paths? Figure 4.13 answers this question by plotting, for each source-destination pair, the maximum fraction of daily traffic transferred over any single path. Only 20% of pairs transfer their entire load over a single path. This suggests that, even if at any time only a few distinct paths are in use, these paths change during the day to adapt to variations in traffic and network conditions. This is confirmed in Figure 4.14, which plots the number of distinct paths used throughout an entire day to route traffic for every pair of source-destination PoPs. In Abilene, all source-destination pairs were using at most 6 paths. In the Tier-1 ISP, 75% of source-destination pairs were using 20 paths or less, but a few pairs were using as many as 40 paths.

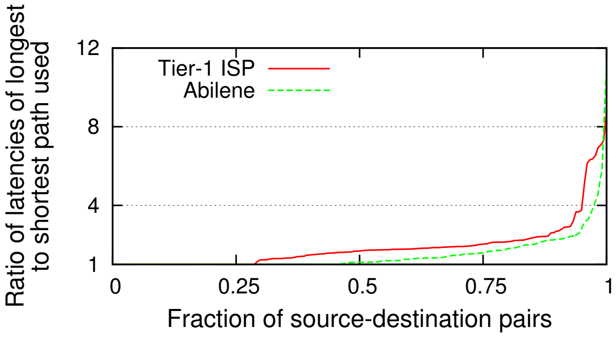


Figure 4.15: Latency stretch of NetEx paths: The stretch is quite high for many source-destination pairs.

These different paths can indeed vary considerably in terms of delay. Figure 4.15 plots, for each source-destination pair, the latency stretch, defined as the ratio of the latency of the longest to the shortest path selected by NetEx during an entire day. The latency stretch can be as high as 10 for some source-destination pairs in our Tier-1 and Abilene networks. Only latency-insensitive bulk applications can afford such a high variability in end-to-end latency.

In summary, even when it routes traffic between the same pair of PoPs, NetEx bandwidth-aware routing selects multiple paths with variable latencies, which suggests that NetEx routing is not suited to interactive applications requiring stable and predictable QoS.

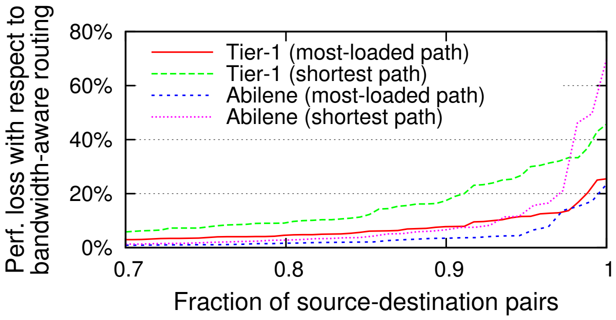
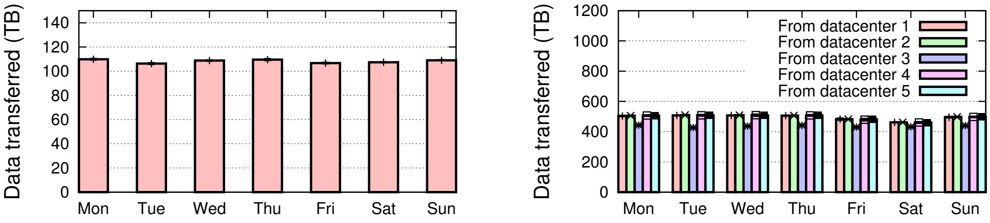


Figure 4.16: Performance loss when using only the most-loaded and the shortest path output by the NetEx optimal routing: Restricting routing to shorter paths result in a larger performance hit.

Having shown how NetEx effectively uses both *multiple* and *long* paths, we now focus on estimating the relative importance of these factors to the performance delivered by NetEx. For this purpose, we take the optimal routing tables produced by NetEx bandwidth-aware routing and filter them to: (1) limit the number of concurrent paths



(a) Full mesh

(b) Single source

Figure 4.17: Performance of NetEx-TE in the Tier-1 European subnetwork with datacenter workloads: Daily data that NetEx-TE can transfer between all datacenters concurrently (a), and from a single data center to all others (b).

used for a given source-destination pair, and (2) reduce their length. We choose to push these constraints to the limit and thus for (1) we allow only a single path, the one carrying the most load, whereas for (2) we allow only the lowest-latency path among the ones selected by the bandwidth-aware routing algorithm. In Figure 4.16 we show that for 80% of source-destination pairs (in both networks), using only the most loaded NetEx path reduces the transferred volume by less than 5%, whereas the maximum penalty is less than 30%. On the other hand, using only the shortest NetEx path has twice as high an impact (80% of pairs sacrifice up to 10%, whereas the maximum loss is around 70%). From the above we conclude that, in the studied topologies, the ability to route over longer paths contributes more to the performance of NetEx than the ability to use multiple paths.

Evaluating NetEx with datacenter workloads

So far we evaluated NetEx using as input the native traffic demands of the ISP. We now evaluate how much data NetEx can deliver under the two datacenter workloads described in Section 4.3.1.

Figure 4.17 plots the daily amount of data that can be delivered by NetEx-TE in the Tier-1 European subnetwork under the two workloads. In the full mesh scenario (Figure 4.17(a)), every data center can deliver at least 100 Tbytes of daily data to all other datacenters [4]. In the case where there is only a single datacenter acting as sender (Figure 4.17(b)), this number rises to nearly 430 Tbytes a day. To put this into perspective, this is more than 100 times the data generated by Facebook picture uploads every day [Fac09]. These results suggest that NetEx has a great potential to serve the increasing traffic generated by applications hosted in datacenters.

[4] For privacy reasons, we had to anonymize the real location of the 5 datacenters.

4.4. Discussion of NetEx design

In this section we address questions related to the deployment incentives of an opportunistic bulk delivery service like NetEx in transit ISPs.

4.4.1. Deployment incentives for transit ISPs

Transit ISPs have an opportunity to monetize resources that are currently being wasted and use them to deliver bulk content. They can lease their spare capacity to offer a lower priority bulk data transit service that does not interfere with their existing Internet transit service. Since such a service would be limited to using only spare resources, transit ISPs could offer it at a lower price than the current Internet service, which is governed by more stringent SLAs that are better suited for delay-sensitive traffic.

Introducing a low cost data transit service could encourage new bulk data workloads. For example, Netflix [Neta] could use the service to move their postal DVD delivery online. Companies could used the service for backing up their data repositories online. Alternately, customers could move some of their existing bulk data traffic to this low cost service. For example, end users could use the service to upload their personal videos and photo albums to social networking sites.

One potential worry is that transit ISPs might hesitate to deploy the service out of fear that customers might divert bulk traffic that is currently being sent over a higher cost service to the new lower cost service. This could happen if the bulk traffic service is priced too low. However, we think that such fears are misplaced. ISPs have entire departments that focus on pricing of their services and a lot of experience in pricing tiered services.

For example, the mobile divisions of many U.S. ISPs offer free “night-time” calls incentivizing users who are price sensitive to avoid expensive “day-time” tariffs by moving most of their calls to night-time. One might similarly worry that such packages could cannibalize the revenue from day-time calls, but, in practice, this is not the case. Day-time calls would still be placed by customers who need to communicate during the day, whereas night-time contracts are there to attract price sensitive customers that would otherwise be unable to afford any kind of mobile service. Thus, by pricing tiered contracts in the right way, mobile operators are increasing their aggregate revenue by attracting customers that would otherwise be lost.

In the case of Internet bulk transfers, the role of price sensitive customers is played by edge ISPs and the role of night-time calls is assumed by bulk transfers (e.g., P2P). Edge ISPs are not willing to pay high rates for all their P2P traffic (as they do for Web, Skype, or gaming traffic) and are therefore rate-limiting such traffic. While finding the right pricing model for NetEx is out of the scope of our work, we believe that if transit ISPs offered a lower cost data transit service at the right prices, edge ISPs would be willing to let more of their bulk traffic get through. This could not only increase the satisfaction of the ISPs’ customers and but also improve the aggregate revenue of transit ISPs.

Design	Traffic isolation	Usage of spare bandwidth
NetEx	Strong	High
Transport-level	Medium	Low
Application-level	Weak	Medium

Table 4.3: Comparison of different designs that use spare capacity to deliver bulk content: The network-level design (NetEx) represents the upper bound on both efficiency and strength of traffic isolation.

Our discussion so far identified several incentives for transit ISPs to deploy NetEx and offer a lower cost transit service to their customers. However, what incentives do transit ISPs have to offer such a service to their peering ISPs that neither pay nor get paid for any data transfers? In this case, NetEx would primarily help to improve the quality of service of interactive traffic by prioritizing it over bulk traffic (that would otherwise compete for bandwidth on equal terms).

4.4.2. Alternatives to NetEx

It may be possible to exploit spare bandwidth in network links with designs that operate above the network layer. In this section, we consider alternative designs to NetEx that work at the transport or application level. We compare them with NetEx along four dimensions: (a) strength in isolating normal traffic from bulk, (b) efficiency in using the available spare bandwidth, (c) deployment incentives, and (d) barrier of deployment. We summarize the tradeoffs in Tables 4.3 and 4.4.

Transport-level design: At the transport layer, protocols like TCP Nice [VKD02] can be used to deliver bulk traffic with minimal interference with interactive traffic. Although TCP Nice can be tuned to keep the interference with interactive flows low, this comes at the cost of efficiency in the usage of spare bandwidth. Making TCP Nice more aggressive increases efficiency, but also increases the negative effects on interactive flows [VKD02].

Application-level design: Another option is to modify applications to schedule bulk transfers at times of low link utilization [LSRS09], exploiting the troughs in the diurnal variation of network traffic. Applications can also make use of overlay networks to reap spare capacity available along multiple paths between a source and a destination by probing for bandwidth and selecting paths that offer higher capacity [ZDA06].

Tradeoffs between the designs

The first comparison criterion is the strength of isolation of normal traffic from bulk. Both network-level isolation and TCP Nice [VKD02] isolate bulk traffic from normal traffic. On the other hand, without OS or network isolation, traffic generated by bulk applications can interfere with normal traffic.

The second criterion is the efficiency in using the available spare bandwidth. Network-level designs can use intelligent traffic engineering techniques to achieve optimal utiliza-

Design	Deployment incentives	Deployment barrier
NetEx	Clear	Changes in hundreds of core routers
Transport-level	Unclear	OS changes in millions of end-hosts
Application-level	Unclear	Software changes in millions of end-hosts

Table 4.4: Deployment incentives and barriers for different designs that use spare capacity to deliver bulk content: The network-level design (NetEx) offers the clearest incentives and is the easiest one to deploy.

tion of spare bandwidth. TCP-Nice on the other hand cannot exploit spare resources available outside the single path between a source and a destination. Overlay routing can exploit bandwidth along multiple paths between a source and a destination. However, since overlay nodes generally select routes to greedily maximize their local performance, overlay routing may fail to efficiently distribute spare capacity among concurrent transfers. In Section 4.3, we found that spare bandwidth usage decreases considerably when each bulk flow is routed independently of other bulk flows (similar to the way overlay routing works).

The third comparison criterion are the deployment incentives. We have already outlined in Section 4.4.1 the incentives for ISPs to offer a lower-priority bulk data transit service, and how users may benefit from it. TCP Nice has less obvious deployment incentives, as it is not clear why users would employ a background transport protocol without incentives from ISPs. Application-level solutions are generally attractive to users. However, ISPs' incentives to support application-level solutions are less clear. In fact, many ISPs today rate-limit overlay-based bulk data transfers [DMHG08].

The final criterion is the deployment barrier of the different designs. The barrier is generally higher for network-level solutions that require updating core routers rather than installing software on end-hosts. However, the number of hosts that need to change in the former case is many order of magnitude lower than in the latter. We also observe that it is often the lack of incentives that discourages ISPs from deploying new designs. With the right incentives, ISPs don't hesitate to deploy even expensive solutions, as evidenced by the billions of dollars residential ISPs like AT&T and Verizon are spending to roll-out high-speed access to end user homes [S., Spa].

4.5. Summary

We compared different designs that exploit spare bandwidth, each design occupying a different layer in the network stack. Our comparison suggests that a network-level design can (a) efficiently isolate normal traffic from bulk traffic, (b) guarantee optimal usage of spare resources, and (c) provide clear incentives for deployment by ISPs. While a combination of transport-layer and application-level solutions may come close to matching the performance of a network-level design (by combining the strengths of each approach), it comes at the cost of higher system complexity and less clear deployment incentives.

We proposed NetEx, a design that can be deployed by transit ISPs without any fundamental changes to their existing infrastructure. We evaluated our design with real data from a large commercial Tier-1 ISP. We found that NetEx increases the amount of data delivered by the network by 60% - 180%, yielding near-optimal usage of spare resources. Finally, we found that bulk flows routed through NetEx achieve a throughput high enough to justify using NetEx for delay-tolerant bulk applications.

Part II.

Home-based distribution of user-generated content

5. Sharing user-generated content from the home environment

Online social networks (OSNs) like Facebook, MySpace, and YouTube have become extremely popular. According to Nielson Online [Nie], OSN sites are visited by 75% of all active Internet households, for an average of 6 hours and 13 minutes a month.

The widespread availability of photo and video cameras, combined with the popularity of OSNs, has made it easy for many people to create, publish, distribute and consume user-generated content. The amount of UGC shared on such sites has grown massively, to the point that one of the primary activities of OSN users today is sharing content with friends (e.g., text updates, web links, photos, videos) [BRCA09]. For instance, Facebook users uploaded more than 15 billion photos to date and continue to upload 220 million new photos every week. In fact, Facebook is the biggest photo-sharing site on the Web [Fac09], demanding 1.5 petabytes of storage and 25 terabytes of additional storage every week.

UGC shared on OSNs is different from other web content. When people publish content on the web, typically their intent is to make the content accessible to Internet users everywhere. In contrast, the intended audience of UGC is often limited. The audience can be explicitly determined by the user or the site's policy (e.g., content can only be seen by friends). Alternatively, the audience can be implicitly limited by the nature of the content. For example, a user's vacation pictures will be of interest primarily to people in the user's social circle. Although most UGC is intended for a limited audience, there are examples of UGC that achieved very high popularity [Youb]. Such UGC is typically public and not protected by any privacy mechanism.

5.1. Current architectures

Despite the fundamental differences between user-generated and web content, OSN users today share UGC using content delivery architectures that were designed for traditional web content. As depicted in Figure 5.1, users upload UGC to centrally managed OSN servers in remote data centers that store the content, often after converting it to a lower-quality format that suits the OSN's storage requirements. Like content uploads, content downloads in OSNs also rely on the traditional web content delivery architecture. For instance, pictures uploaded to Facebook are delivered to users by the Akamai content delivery network (CDN), whose caches are deployed over geographically diverse regions, for example to provide satisfying response times to users.

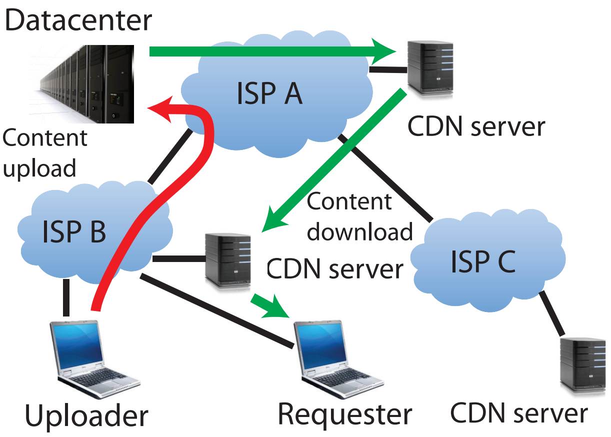


Figure 5.1: Current architecture for sharing content in OSNs.

5.1.1. Drawbacks of the current architecture

While the traditional web infrastructure scales well, it has several disadvantages when it is used to deliver UGC [SVCC09, LB08]. Several aspects contribute to these disadvantages, including:

1. **Constraints on content shared** OSN users sharing personal data are often subject to various site-specific constraints. Some sites allow particular types of contents to be shared but not others (e.g. Facebook and Flickr allow pictures and videos but not music). OSNs like Facebook and YouTube constrain the size and the resolution at which multimedia content can be shared.

2. **Ownership and copyrights** Users who upload personal content to OSNs are often subject to complex (and dynamically changing) terms of ownership rights. For example, many OSNs like Facebook demand fairly broad rights to use the content shared on their sites [FACc].

3. **Privacy** The last but perhaps most widely recognized concern with sharing data using OSNs is the associated loss of privacy. OSNs are known to change their privacy settings for uploaded content in ways that often catch ordinary users off-guard and compromise the privacy of the data they share [faca, SZF10].

From the perspective of the OSN provider, managing the deluge of UGC is becoming increasingly challenging and expensive [Fac09]. The traditional web delivery infrastructure is optimized to serve highly popular content that lends itself to performance improvements through CDN caching. UGC, however, is often of interest to a small audience and hence unlikely to become very popular. In fact, in Section 5.5, our study reveals that up to 97% of all photos shared over Flickr and 50% of all videos shared over YouTube are never accessed during the course of a given week. This translates into huge amounts of wasted storage capacity in data centers. Furthermore, the 3% of photos and 50% of videos that are accessed are only requested a small number of times, which reduces the effectiveness of CDN caching. Of course, the difficulty of managing and delivering UGC from millions of users is offset by the benefit of using this personal data for profitable activities such as targeted advertising. However, this raises further privacy concerns as users rarely know how their data is exactly being used and what third parties have access to it.

5.1.2. An alternative approach: delivery of UGC from home

Letting users deliver UGC directly from their homes is attracting a lot of attention as an alternative to current centralized solutions [BSVD09, SVCC09, diab, Diaa]. Home-based UGC sharing is becoming more practical due to recent trends such as the availability of large, inexpensive home storage devices and always on, high-speed broadband connectivity. With home-based sharing, users can regain control over their UGC. Finally, because most UGC is generated by users in their homes, home-based sharing eliminates the need for uploading content to remote data centers.

Despite its appeal to end-users, it is still unclear how home-based UGC sharing would work in practice. Unlike centralized infrastructures like data centers and CDNs that are well provisioned and well managed by expert operators, home networks have limited resources (both storage and bandwidth) and are managed by lay users. Consequently, there are several unresolved concerns related to the security, availability and performance of home-based UGC sharing.

5.2. Contributions to help users regain control over UGC

We make the following contributions towards enabling delivery of UGC directly from users' homes:

Study of the feasibility of UGC delivery from home We present a measurement-driven feasibility study of home-based UGC sharing. To conduct this study, we first needed to understand (a) *the characteristics of OSN workloads*, i.e., patterns of UGC uploads and downloads, and (b) *the characteristics of home networks*, i.e., the availability and utilization of residential access links. To this end, we gathered and analyzed detailed real-world traces from OSNs and home networks. Later, we used these traces to analyze the extent to which UGC can be stored and delivered from users' homes.

More specifically, we make three contributions:

1. **Characterizing OSN workloads** We gathered data from two popular OSN sites (Flickr and YouTube) to understand the nature of OSN workloads.
2. **Characterizing home networks** We deployed a distributed testbed in 10 households located in 2 continents and 9 ISPs. We used this testbed to monitor the Internet usage patterns of real users, as well as measure the availability and performance of distributed content delivery from home network environments.

3. **Evaluating the feasibility of storing and delivering content at homes** Through simulations, we estimated the extent to which the workloads from our OSN workload study can be served using the resources available in the home environments we measured.

Stratus: a prototype system that delivers UGC from users' homes To demonstrate that home-based sharing of UGC is indeed a practical alternative, we designed and implemented Stratus, a system that enables users to share their personal content with friends and acquaintances directly from home. Our implementation of Stratus runs on currently available wireless routers that users can employ as their Internet home gateways. These routers are equipped with a USB port, and can therefore be connected to commodity mass storage devices. Stratus makes some content stored on these devices available to other users whom the content owner has granted access permissions to. We also implemented a Facebook application that enables Stratus users to easily share personal content from home with any of their Facebook friends, thus demonstrating how Stratus can be integrated with existing online social networks.

5.3. Related work

Recently, a number of alternative UGC distribution systems have been proposed. For instance, PeerSon [BSVD09] sketches a social network that runs on a peer-to-peer (P2P) network, where individual users manage their own storage and use a distributed hash table (DHT) to locate content. In addition to purely distributed designs, there are also hybrid designs that utilize home networks as well as centralized infrastructure. One such example is the Vis-a-Vis [SVCC09] system, which relies on users' desktop machines and the cloud computing architecture to exchange content. Another example is Diaspora [diab, Diaa], a recent project that aims at creating a fully-decentralized OSN entirely controlled by end-users. Diaspora is a personal web server that manages the user's personal information, like OSN contacts and shared content. Diaspora personal servers can also be located directly at users' homes.

This work investigates the feasibility of delivering UGC from users' homes, and is motivated by the emergence of always-on home-based storage systems, such as PogoPlug's external hard drive [Pog] and NetGear's ReadyShare wireless routers with USB storage [Netb]. These technologies allow users to share content, including video and full resolution photos, with friends and family without having to upload the content to OSNs.

5.4. Alternative decentralized architectures for home-based UGC sharing

Home networks can be used for sharing UGC in different ways. Below, we describe three broad types of system designs that we later evaluate:

1. **Personal home-server system:** In this purely decentralized system, all content requests are served directly by the user who shared the content (i.e., the publisher). Content is delivered at the time of a request and users who request the content (i.e., consumers) are not required to own home servers. In this system, publishers own and thereby retain full control of the infrastructure (i.e. the home servers) used to share their content.

2. **Hybrid system:** In this architecture, *very popular* content is served with help from a third party (e.g., either centralized, well-provisioned data centers or by resources contributed by the publisher's friends), while the remaining content is served directly from the publishers' homes. This system is similar in spirit to CoralCDN [FFM04], where a web site redirects requests for "hot" content to a scalable CDN.

3. **Push-based system:** In this purely decentralized system, content is pushed to the target audience as soon as it is published. In order to store content, both publishers and consumers must own home-based storage devices. By prefetching content before it is actually requested, push-based systems can not only cut down the access latencies for content consumers but they also relieve the publishers from being overwhelmed with multiple simultaneous requests for content. Given the limited bandwidth resources in a home environment, even a small number of simultaneous requests can congest the networks. The drawback of the push-based system is that it requires time to push the content to the target audience. Therefore, it still cannot make new content available quickly to a large number of people.

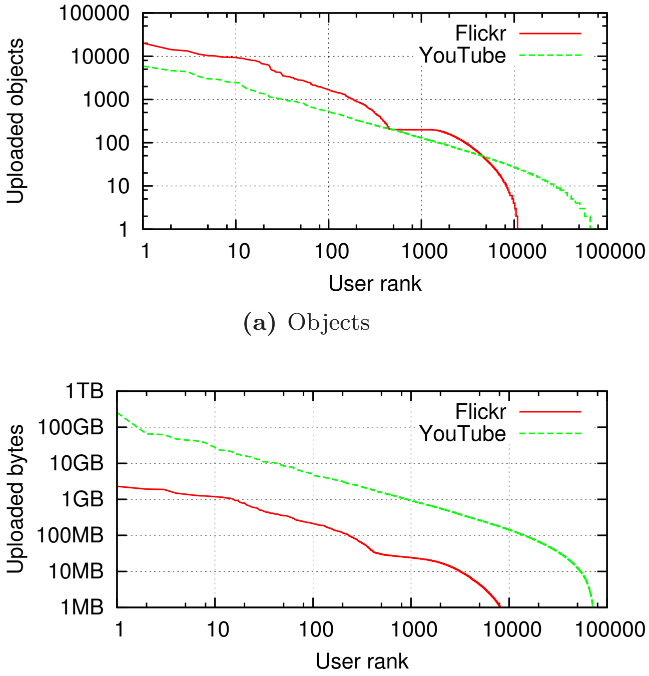
5.5. Understanding UGC workloads

Websites that allow users to upload and share user-generated content, in particular OSN websites, have changed the way in which content is distributed on the Internet. To highlight the differences between UGC and the traditional web content, in this section, we use real traces gathered from popular OSN sites and study their characteristics.

5.5.1. Datasets

We implemented a web crawler for `flickr.com` and `youtube.com`, two popular sites that allow people to share user-generated content with their friends. Our crawler gathered detailed information about the uploads and downloads of publicly available content from these sites.

- **Flickr:** We randomly selected 11,715 users from the list of 2.5 million users gathered by [CMG09]. We crawled the profile pages of these users daily for 19 consecutive days. We then collected information about all 1,324,080 public photos



(a) Objects

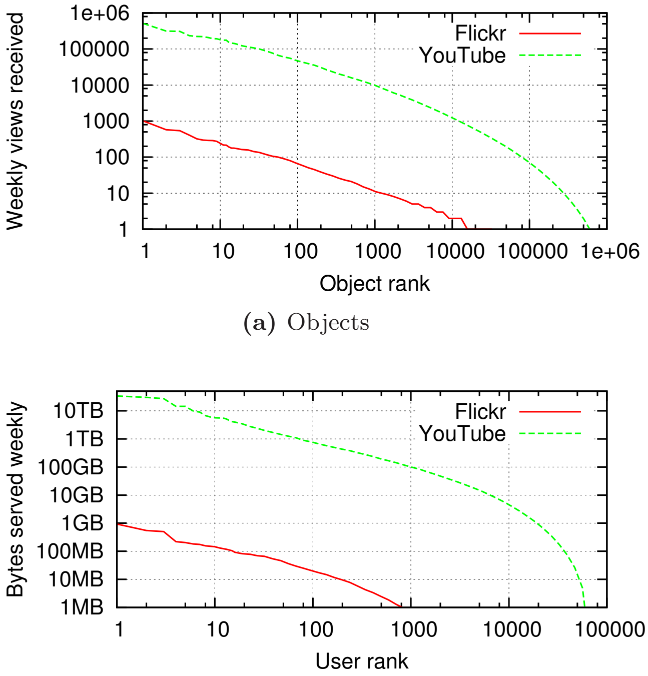
(b) Bytes

Figure 5.2: Content production pattern shown as the rank distribution of users and the number and total size of objects they uploaded.

uploaded by these users. For each photo, we recorded the number of daily views received, as well as metadata, like photo size, tags, and favorite markings.

- **YouTube:** We randomly selected 77,575 users from the list of YouTube users gathered by [CKR+07]. We collected information about the public videos uploaded by these users, for a total of 1,251,492 videos. We collected the number of daily views for all these videos over a period of 166 days using the "StatisticsAndData" feature in YouTube.

Ideally we would have also liked to include data from other OSN sites like Facebook, but obtaining data from such sites is hard because most of the content shared is private. In contrast, all the data we gathered from Flickr and YouTube is publicly accessible. Furthermore, these two sites provide mechanisms for searching and featuring popular content. Hence, our analysis of content *consumption* patterns is likely to overestimate the popularity that content would have reached had it been shared on OSN sites like Facebook.



(a) Objects

(b) Users

Figure 5.3: Content consumption pattern: (a) object ranks and the number of weekly requests and (b) the total amount of content served by Flickr and YouTube on behalf of content uploaders.

5.5.2. Content production patterns

In order to understand the storage requirements for sharing UGC content from home, we study the content production patterns of Flickr and YouTube users. We examine the total amount of content shared by each user in our dataset since they joined Flickr and YouTube. Users in our dataset joined the OSN websites 4 years ago on average.

Figure 5.2(a) shows the rank of each user against the total number of objects (photos and videos) they shared, in a log-log plot. Flickr shows a plateau at 200 pictures, a consequence of the limit imposed on the number of photos visible in a free account. The content production rate is generally low; users uploaded on average 111 photos (median=29) and 16 videos (median=6). Only 10 Flickr users (accounting for 0.08% of all users) uploaded more than 10,000 pictures. Likewise, only 40 YouTube users (0.05%) uploaded more than 1000 videos.

Figure 5.2(b) shows the same trend as a function of the total size of uploaded content. Because we are interested in the storage requirements for active users, we only show users who uploaded more than 1 MB. While a small fraction of users uploaded more than 100 GB of videos, the remaining users' uploads remain small in size. On average, users uploaded 13.3 MB to Flickr and 103 MB to YouTube. Even the most prolific user on Flickr uploaded less than 3 GB. This amount of data can easily fit into a small storage device, e.g., a USB-stick attached to a home gateway. Our analysis indicates that while the total amount of content that is shared by all users on OSNs is massive, *individual users only share a limited amount of content* and this content can fit on affordable storage devices.

5.5.3. Content consumption patterns

We not turn to UGC consumption patterns in order to understand how frequently UGC content is requested. We examine the number of requests each shared object and each uploader received in a typical week. Due to space limitation, we present the request patterns based on the last week of our data. However, we did not observe significant changes when we examined other randomly chosen weeks.

Figure 5.3(a) shows the number of requests each shared object in Flickr and YouTube received during one week. We make three observations.

First, YouTube videos receive in general more requests than Flickr photos. Many factors may contribute to this difference such as the difference in the popularity of the two sites—according to `alexa.com`, 22% of global Internet users visit YouTube, while only 2.5% visit Flickr.

Second, not all 1,324,080 Flickr photos and 1,251,492 YouTube videos were requested during a week period. Rather, *a substantial fraction of objects did not receive a single request* during an entire week. More precisely, 97% of Flickr photos and 50% of YouTube videos were never requested during the one week period. These results suggest that the number of objects that need to be made readily available through web servers and CDNs can, at least potentially, be drastically reduced.

The second observation is that even the content that was requested received only a few requests during the one week period. Almost all Flickr photos received less than 1,000 requests. YouTube contained about 1,000 very popular videos, which were viewed over 10,000 times. However, 94% of all videos (or 88% of all videos with at least one request) received no more than 100 requests. The fact that many objects are unpopular is promising for the feasibility of a decentralized architecture, because it reduces the resource demand on home networks.

Finally, in order to see how requests for objects are distributed across users, in Figure 5.3(b), we show the total number of bytes that Flickr and YouTube served on behalf of the uploaders. This number is important because it represents how much data users would need to serve from their homes. 90% of YouTube users need to serve 7.6 GB/week or less. This corresponds to an average bandwidth of 100 Kbps, and is thus a demand that home connections can potentially meet. The bandwidth demands for Flickr photos are roughly two orders of magnitude smaller. With respect to latency, our analysis in Section 5.5.3 suggests that requests for personal content tend to come from the same geographical region as the uploader's. If this is true, the latency experienced by the requests is likely to be small.

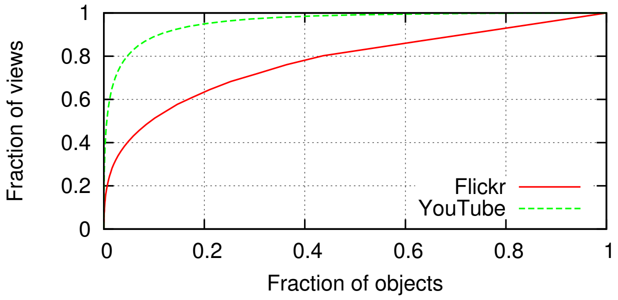


Figure 5.4: How many of last week's requests do the top-objects account for?: In Flickr, it is necessary to cache as much as 40% of the top objects to satisfy 80% of all requests.

How cachable are UGC workloads?

Among the object that receives at least one request during our sample week, how many of them accounted for most of the request? Answering this question can give some useful insights on the cachability of UGC workloads. Traditional web caches work well because web content tends to have a small number of very popular objects that account for most of the request. Does this also happen in OSN workloads? Figure 5.4 plots the fraction of objects versus the fraction of requests they account for. Flickr photos have poor cachability: one would need to cache 40% of all requested photos to account for 80% of the requests. In the case of traditional web content, caching 20% or 10% of objects suffices. YouTube videos on the other hand present excellent cachability. We believe this is because many YouTube users upload content of general interest that tend to become very popular, such as music videos or movie trailers.

How local are UGC workloads?

How much locality is there in UGC workloads? Are requests for content mostly coming from users located in the same country/region as the content's uploader's? In order to answer this question, we would ideally have recorded where all requests for photos and videos in Flickr and YouTube come from. Since this information is not available, we had to resort to analyze comments about YouTube videos and favorite markings of Flickr photos. This is because comments and favorite markings contain information about the user who created them, and thus allows us to determine their location and compare it with the uploader's. Also, because a user must view a piece of content in order to mark it or comment on it, favorite markings and comments are a lower bound on the number of views that the content received, and represent a reasonable estimate of its popularity.

In our dataset, 62% of YouTube videos have comments, but less than 5% of Flickr photos have favorite markings. We therefore focus on YouTube comments.

Table 5.1 reports the fraction of comments from users located in the same country as the video's uploader's, for videos with different numbers of total comments. We

	Comments	
Videos with	**Total**	**Same country**
1 comment	109K	64%
2-10 comments	1.1M	57%
11-50 comments	2.7M	50%
51-100 comments	1.8M	49%
>100 comments	20.1M	53%
All videos	26.5M	53%

Table 5.1: Fraction of YouTube comments from users located in the same country as the video uploader's: The count is restricted to comments from users for which a location could be inferred. There were 26.5M (66%) such comments, from a total of 39M. comments

ignored comments from the video's uploader. Across all videos, a majority of comments (53%) come from the same country as the uploader's. If we restrict ourselves to low-popularity videos (e.g. 10 comments or less), the fraction increases. These results seem to indicate that requests for content in YouTube tends to be local, at least at a country-level granularity. This has interesting consequences for the distribution of UGC content. Most UGC content today is uploaded to remote datacenters, often located on a different continent, and distributed from there. However, if it turns out that most requests come from users located near the uploader, it would make sense to consider a distributed system that keeps the transfers local. This would reduce the datacenters' costs as well as the inter-ISP traffic [1].

5.6. Characterizing home network environments

Compared to well-provisioned and professionally-managed centralized infrastructures like data centers and CDNs, most home network environments are resource-constrained and are managed by lay users. This raises concerns about the reliability of sharing UGC from users' homes. In this section, we quantify the reliability of home network environments based on real measurements.

5.6.1. Methodology

For this work, we built our own home servers using NetGear wireless routers and deployed them in a number of households.

[1] Since ISPs' backbone boundaries often coincide with country boundaries, country-level locality may be enough to ensure this.

Design of our home servers

We implemented our home server on a NetGear WGT634U home router [Netc]. This router is equipped with a 200 MHz MIPS CPUs, and runs OpenWrt [Ope], a Linux distribution for embedded devices. The router also runs a lightweight HTTP server [lig] to serve content stored on a 2 GB USB drive attached to the router, as shown in Figure 5.5.

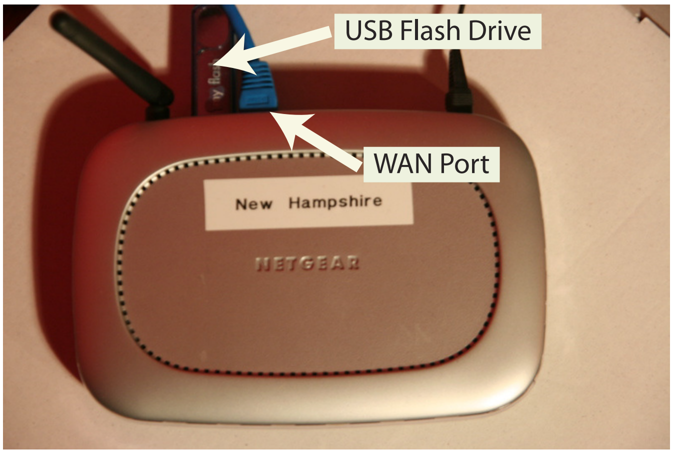


Figure 5.5: Wireless router equipped with USB storage used in testbed.

Our home servers are inexpensive and require only a limited amount of power. The cost of a home router comparable to the one we used is around $60. The cost of a 2 GB flash drive today is $9. If additional storage is needed, it is possible to attach external USB hard disks that provide several hundred gigabytes or even terabytes of storage space. The wireless router is powered by an adapter whose maximum power output is 12 watts. Even assuming constant maximum power consumption, the router would consume about 100KWh over an entire year, which translates into a yearly cost of $18 (using electricity retail prices in the New York area in February 2010 [ELE]). Hence, we claim that an always-on home server is an affordable solution for most users. There might be a concern that an always-on device is not environmentally friendly. However, we are not proposing to add a new device to the home environments, but rather to use an already existing, always-on device (the home gateway) for an additional task (content distribution).

Measurement testbed

We deployed our home gateways in 10 households in 2 different continents: Europe (Germany, Spain, and Italy) and Asia (Korea). The gateways are connected to 9 different Internet Service Providers (ISPs). The data from these gateways was collected over 79 consecutive days from April 1st to June 18th, 2010. In all cases, our gateway was being used as the primary source of Internet access by the people living in the household. This enabled us to monitor the amount of Internet traffic on each Internet connection as well as the number of local devices present in the network.

On each router, we ran a measurement daemon that performed the following three tasks: (1) sending minute-by-minute heartbeat messages to a remote tracking server, in order to infer the availability of gateways; (2) monitoring all traffic sent through the gateway, in order to measure utilization of the residential Internet links and detect the presence of any local devices accessing the Internet; and (3) periodically fetching media files (pictures and videos) from randomly selected routers in the testbed as well as from Facebook, in order to compare the performance of a decentralized architecture with that of Facebook. The media files exchanged and the logs of all the results were stored on the USB storage devices.

While the number of deployment instances of our testbed might seem small, especially when compared to prior studies of residential networks [DHGS07, LPP04], no prior study has ever gathered such detailed data over several months and performance measurements about home network environments as we do from each of our home deployments. Such data is necessary to evaluate the content delivery capacity of home networks.

5.6.2. Availability of home gateways

We used data from the heartbeat messages to infer the availability of gateways. We consider a router to be *unavailable* if the tracking server misses five consecutive heartbeat messages from the router, i.e., does not hear a heartbeat over a period of five minutes. By waiting for five consecutive message losses, we reduce the chance of misinterpreting occasional packet losses as router unavailability.

Table 5.2 reports the availability of home gateways. Overall, the availability of gateways is generally high—around 98%. Unavailability periods are typically short; the median unavailability period is just 11 minutes. Occasionally, the unavailability periods lasted from several hours to a few days. Anecdotal evidence suggests that this happened when users turned off the power and left their home for a long time. The longest unavailability period lasted 3.6 days. However, in 90% of cases, the unavailability lasted less than 12 hours.

Average connected time	98%
Median unavailability period	11 minutes
Unavailability period (90th percentile)	12 hours
Longest unavailability period	3.6 days
Average # of disconnections per day	0.1
Average # of IP changes per day	0.4

Table 5.2: Statistics about the availability of monitored gateways.

Another potential cause of unavailability are ISPs periodically resetting the home Internet connection to reassign the IP address of home gateways. However, Table 5.2 shows that IP changes were infrequent. In fact, this happened for only 1 of the 9 ISPs we monitored. Also, when the connection was reset, the loss of connectivity lasted significantly less than five minutes and was thus never registered as an unavailability period.

5.6.3. Availability of home devices

How would the availability change if, instead of home gateways, we used laptops and desktop computers as servers? To understand the availability of these home devices, we measured how long home devices are connected to the gateways. On average 3.1 different local devices were connected to each gateway at some time. While most home networks had multiple devices, 73% of the time there was no device connected to the gateway. In fact, even the most available local device (i.e., the device that remained connected to the gateway the largest fraction of time) was connected only 62% of time. The availability of home devices compares poorly with that of gateway servers (with average availability of 98%). This suggests that serving content directly from home devices might not be a viable solution.

5.6.4. Utilization of home access links

Residential Internet access links are known to have limited capacities [DHGS07]. Furthermore, a home gateway server can only rely on access link bandwidth that is not being used by home devices. A crucial question therefore is *how often are access links of home networks utilized and to what extent?*.

To answer this question, we analyzed the data collected from monitoring home network traffic. We computed the average utilization of all links over each 5-minute interval and plotted it in Figure 5.6. The upstream links are unused more than 80% of the time, while the downstream links are unused more than 40% of the time. Furthermore, for 95% of the time, the link usage was below 15 Kbps and 230 Kbps in the upstream and downstream directions, respectively. We also looked at the hourly usage of individual upstream links and found that usage is very bursty and very low (below 50 Kbps) when averaged over one-hour periods. Since all the access links had a downstream capacity of several Mbps and an upstream of several hundreds of Kbps, the results show that even when the access links were being used, they had plenty of spare capacity left for other traffic.

5.6.5. Content sharing performance

In order to assess the performance of sharing content from home gateways, we stored 20 JPEG pictures and 1 MPEG4 video file on the USB drive of each gateway and measured the performance of fetching the files from other gateways. For comparison purposes, we uploaded the same media files to Facebook. The size of the files were between 80 KB and 130 KB for pictures and 18 MB for the video.

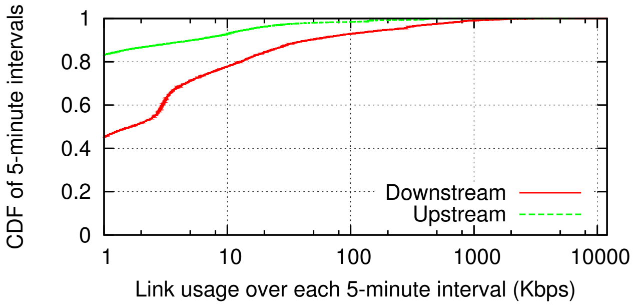


Figure 5.6: Utilization of residential access links (all gateways).

Every 10 minutes, each gateway requested the pictures from a randomly chosen gateway and from the Akamai URL [2] used by Facebook to deliver the files. The same was done for the video file, although only once every hour. For each download, we recorded the completion times and any error and HTTP response codes.

Successful content downloads

We discuss how often the media file downloads were successfully completed. Table 5.3 displays the statistics for the content downloads. The percentage of successful downloads is quite high both using home servers and Akamai (93% using home servers and 99.7% using Akamai), although Akamai is clearly preferable if one needs a highly reliable service.

Table 5.3 also reports the major sources of errors that caused content downloads to fail. The major sources of error for Akamai were failed DNS resolutions, where the client could not successfully resolve the Akamai URL. In the case of content served from the testbed, the major sources of errors were internal server errors and empty responses. After inspecting the logs, we found that a lot of these errors were generated by a single gateway with faulty USB storage. Excluding this outlier, the main source of error was failed connections to the server. This accounted for 1.8% of the cases, which well matches the 98% availability of the gateways presented above.

Performance of photo browsing

Next we look at the time taken to complete the photo downloads. Table 5.4 reports the percentile of download times in the experiments. Even when photos were served from home gateways, 80% of the downloads took less than 3 seconds, and half of the downloads took less than 2 seconds. In order to hide fetch latency from the user, photos in the same photo album could be prefetched in the background. Prefetching seems to be useful because users, unless they are browsing quickly, are likely to spend a few seconds viewing a photo before requesting the next one.

[2] Before every transfer, the gateway resolved the Akamai URL with a DNS query to obtain the current Akamai server's IP.

	Served by Akamai			Served from home		
Transfer outcome	**Photo**	**Video**	**All**	**Photo**	**Video**	**All**
OK	99.8%	98.7%	99.7%	93.1%	82.8%	93.0%
Not found	0.001%	0.9%	0.01%	0.4%	0%	0.4%
Server internal error	0.0002%	0%	0.0002%	2.4%	2.6%	**2.4%**
Empty response	0.003%	0%	0.003%	2.0%	2.4%	**2.0%**
Connection failed	0.02%	0.1%	0.02%	1.8%	2.0%	**1.8%**
DNS resolution failed	0.2%	0.2%	**0.2%**	0.01%	0.02%	0.01%
Total	1,517,406	12,521	1,529,927	1,060,027	8,700	1,068,727

Table 5.3: Summary of content download outcomes: Akamai's failed transfers are dominated by DNS resolution errors, whereas failures in the testbed are dominated by a single faulty gateway, and failed connection attempts due to disconnected gateways.

	Photo download time (sec)	
Percentile	**Akamai**	**From home**
10th	0.11	0.58
50th	0.36	1.91
80th	0.81	2.91
95th	1.38	5.32
99th	4.69	10.33

Table 5.4: Time required to download a photo from Akamai and our testbed.

Performance of video streaming

Unlike photos, which are typically looked at after being downloaded, videos are often watched as on-demand streams. Therefore, when evaluating the performance of video sharing, we looked at download bandwidths rather than download completion times.

Figure 5.7 reports the average bandwidths achieved during the media streaming experiments. The testbed cannot compete with the performance of the Akamai servers. However, 95% of transfers achieve an average bandwidth higher than 200 Kbps (which correspond to low-bit rate streams), while 66% of transfers achieve an average bandwidth higher than 400 Kbps—an encoding rate that is higher than a majority of YouTube videos [GALM07]. The transfer bandwidths are by and large limited by the upstream capacities of home Internet connections.

High average bandwidth alone does not guarantee that video streaming was uninterrupted. To understand whether a user would be able to watch a video streamed from home servers without interruption, we recorded the bandwidth achieved in every 1-second interval of streaming downloads and used the data to compute how many playbacks with a certain encoding rate would complete uninterrupted. In the computation, we assumed that all videos have a duration of 140 seconds. We also considered different pre-buffering times (i.e., the time between the begin of the video download and when the first frame is shown to the user).

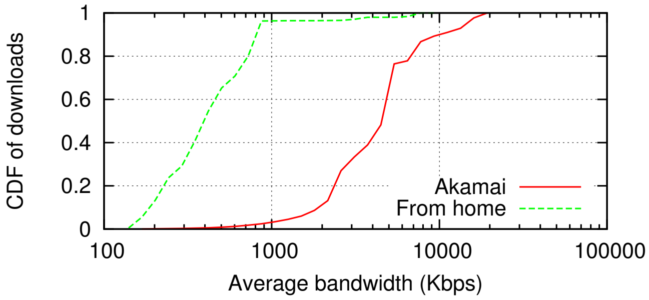


Figure 5.7: Bandwidth achieved by video downloads from Akamai and our testbed.

Figure 5.8 shows the fraction of uninterrupted media playbacks for gateways whose upstream capacity is at least as high as the streaming bit rate. For low bit rates (100-200 Kbps), two seconds of pre-buffering are sufficient for most playbacks to end without interruptions. These bit rates are more than enough for high-quality MP3 audio files, thus showing that music can be effectively streamed from home. For YouTube-like bit-rates (400 Kbps), a considerable amount of pre-buffering is needed to lower the fraction of uninterrupted playbacks. For example, if the content is pre-buffered for 5 seconds (equal to 3% of the video duration), almost 80% of playbacks succeed. At higher bit-rates, no reasonable amount of pre-buffering can reduce the fraction of interrupted playbacks.

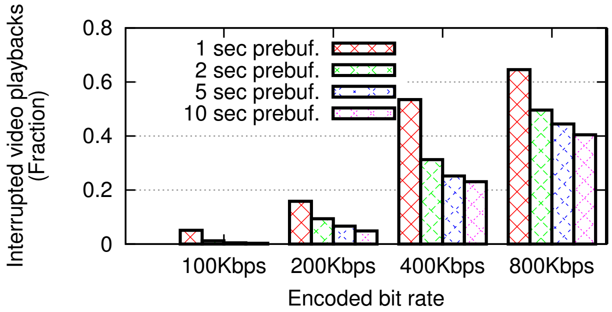
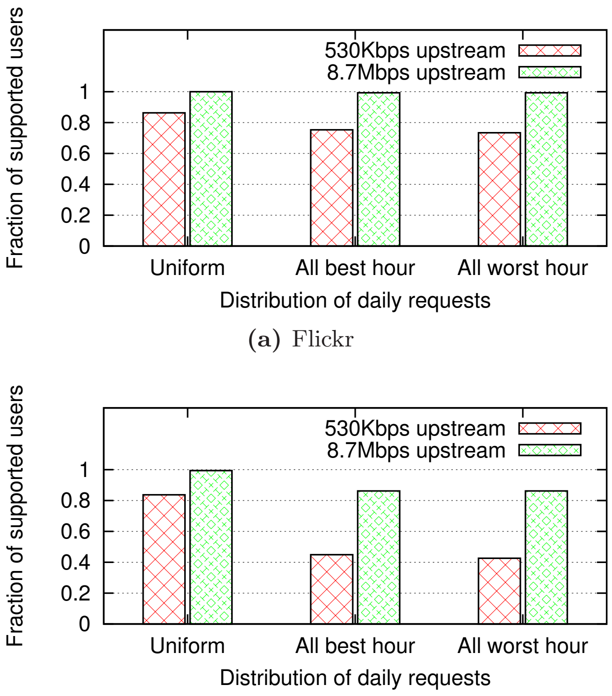


Figure 5.8: The fraction of interrupted playbacks across varying pre-buffering times and encoding rates.

5.7. Potential for sharing content from the home

So far we have collected data on UGC workloads and measured the availability of resources in home environments. We now combine these observations to evaluate the potential for delivering UGC in a decentralized manner.



(a) Flickr

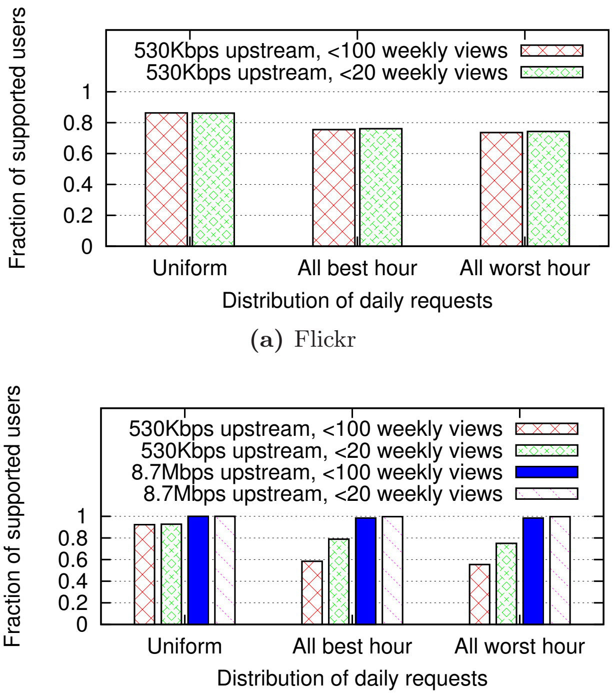
(b) YouTube

Figure 5.9: Fraction of users who successfully delivered 95% of requests from home gateways in a home-server system.

5.7.1. Simulation setup

We implemented a simulator to estimate the fraction of UGC requests that can be served by each of the system designs sketched in Section 5.4: a personal home-server system, a hybrid system combining home servers with some third-party infrastructure (either well provisioned servers or home networks of friends), and a push-based system. The simulator takes three types of input from our measurements: (1) the number of daily requests for each user's objects (photos and videos) from the last week in our Flickr and YouTube dataset, as well as the size of the object; (2) the spare capacity in the upstream link at each home gateway during the last week of our measurements; and (3) the time periods during which the network gateways were available (i.e., could be reached from the Internet).

We use input (1) to drive the arrival of requests for each content publisher in the simulator. Because we only have information about the aggregate number of requests per day, we determine the precise arrival time of each request based on three different distributions: First, a uniform distribution, where requests arrive uniformly throughout the day; second, a best-case distribution, where all requests arrive during the hour of the day in which the gateway has the most spare capacity; third, a worst-case distribution,



(a) Flickr

(b) YouTube

Figure 5.10: Fraction of users who successfully delivered 95% of requests from home gateways in a hybrid system.

where all requests arrive during the hour of the day in which the gateway has the least spare capacity.

We use inputs (2) and (3) to determine whether a request succeeds or fails when served by a given gateway. Upon every request arrival, we first check if the gateway is online based on input (3). If the gateway is online, we use the information from input (2) and examine if the spare capacity is enough to satisfy the request. We consider a content request a *success* if the download completes within 3 seconds for photos and if the average download speed is at least 400 Kbps for videos. Otherwise, we consider a request a *failure*.

For each content publisher in our Flickr and YouTube datasets, and for each of our gateways, our simulator estimated the fraction of requests for the publisher's content that could be served by the gateway. Below, we present results for two of our home gateways: the one with highest upstream capacity (8.7 Mbps), and the one with the median upstream capacity (530 Kbps). These two gateways represent two characteristic points in the spectrum of currently available broadband connection technologies. The higher upstream capacity of 8.7 Mbps represents users with fast broadband connections like those in South Korea and Japan [Lei09, Us]. The median upstream capacity of 530 Kbps conservatively represents users with typically slower broadband connections like those in the United States and Europe [DHGS07]. Recent data on the penetration of broadband Internet connections [Lei09] suggests that a majority of users worldwide might have access to similar capacities in the near future.

5.7.2. Personal home-server system

We first discuss the results for the personal home-server system, where the home gateway of the content publisher directly serves all requests. Figure 5.9 shows the fraction of users for whom at least 95% of the weekly requests are successfully served[3] from gateways, for the three different request distributions. If requests are spread uniformly throughout the day, even the median-bandwidth gateway can support more than 80% of both Flickr and YouTube users. A high-bandwidth gateway can support all users. For Flickr, these results don't change much if all requests arrive within an hour. For YouTube however, if all requests arrive within an hour the fraction of YouTube users supported drops considerably for both the high (from 99% to 86%) and median (from 83% to 42%) bandwidth gateways.

For Flickr, the main reason why some users cannot be supported is that the upstream capacity is insufficient to deliver some of the large photos. While the average size of a Flickr photo is 120 KB, a small subset of photos (4%) are larger than 200 KB, and thus cannot be delivered in 3 seconds on a 530 Kbps link (530×3/8≈200 KB). For YouTube, however, the main limiting factor is multiple requests competing for the same upstream bandwidth. For instance, when we assume that all daily requests arrive within an hour, a considerable fraction of users have to serve more than 20 video requests in the same 5-minute interval, which requires an aggregate bandwidth of 8 Mbps (400 Kbps×20) or more. Therefore, even with a good upstream link, not all requests could be handled.

5.7.3. Hybrid system

In the hybrid system, the publisher offloads popular UGC to a third-party infrastructure, e.g., well-provisioned, centralized OSN servers or resources contributed by the publisher's friends. Thus, the publisher's personal server only has to serve requests for the remaining, less popular objects. While object popularity could be inferred based on the past history of views or the intended audience, the detailed techniques for determining popular objects are beyond the scope of this work. Therefore, we restricted the home gateways in our simulator to serve objects with a limited number of weekly requests.

Figure 5.10 shows the fraction of supported users when the home gateway only serves objects that received X or fewer requests in a week. We show the result for X=20 and 100. To give an idea of how many users are affected by these thresholds, we observe that 44% of YouTube and 4.6% of Flickr users uploaded at least one object that received more than 20 weekly requests.

We first focus on the results for Flickr shown in Figure 5.10(a). We omit the results for the high-bandwidth gateway, because we saw that the high-bandwidth gateway can

[3] We believe a 95% success rate is sufficient for an application like OSN sharing. For completeness, we repeated the experiments requiring a 100% success rate and did not find a large difference in the results.

already support all Flickr photos for all users even with the simple home-server system. With the median-bandwidth gateway, the hybrid system does not bring a noticeable performance improvement. This is because the size of individual photos, and not contention for capacity from multiple requests, was the main factor that limited performance. In fact, 99.9% of the Flickr photos received less than 20 weekly views.

For YouTube however, the hybrid system achieved considerably better performance (Figure 5.10(b)). When home gateways only serve videos with 20 weekly views or less (X=20, accounting for 84% of all videos), the fraction of supported users for the median-bandwidth gateway goes from 42% to 75%. In the case of the high-bandwidth gateway, the fraction goes from 86% to 99%.

5.7.4. Push-based system

The push-based system pushes any published content to a subset of the publisher's friends who are likely to request it in the future. Hence, future requests for UGC are immediately satisfied by the requester's gateway. Such feature of the push-based system is similar to speculative prefetching [KLM97], where content that is likely to be requested in the future is downloaded in advance.

In order to test the feasibility of the push-based system, we use the content upload history of users and the number of contacts each user has in YouTube and Flickr. On average, our Flickr and YouTube users have 14 and 15 contacts, respectively. We ran our simulation to see if the content uploaded on each day by a user can be delivered to the user's contacts within a specified amount of time—e.g., a day. For the sake of completeness, we also considered a more challenging scenario where each user is assumed to have 130 contacts, which is the average number of friends in Facebook [FACb].

Table 5.5 shows that the vast majority of users can indeed push UGC to their friends even using a median capacity gateway with only 530 Kbps upstream bandwidth. Most users could still deliver content successfully even with a relatively high number of contacts (130). These results suggest that push-based systems can help home-based UGC sharing architectures reduce both content access latencies for content requesters and bandwidth contention on upstream links for content publishers. The drawback of push-based systems is that they cannot quickly make content available to many people because they require time to upload the content to the potential requesters.

	Actual # of contacts		**130 contacts**	
Upstream	**Flickr**	**YouTube**	**Flickr**	**YouTube**
530Kbps	99.8%	98.8%	99.4%	91.2%
8.7Mbps	100%	99.9%	100%	99.9%

Table 5.5: Fraction of users whose daily published content could be pushed to all OSN contacts within one day in a push-based system.

5.7.5. Summary

We summarize our main findings. First, the workload analysis of the Flickr and YouTube social networks in Section 5.5 showed that most OSN users upload only a small amount of content and that most uploaded content has limited popularity. Thus, UGC workloads of individual users are not very demanding, which bodes well for supporting them from resource-constrained home environments. Second, our measurement analysis in Section 5.6 highlighted that residential connections have low utilization and good availability, but that in order to harness this availability an always-on home gateway is required, because local devices tend to disconnect frequently. Finally, the simulation results in Section 5.7 suggest that, while not all UGC can be delivered from the home with good performance today, the current trends in the capacity of residential connections bode well for home-based UGC delivery. Moreover, alternative designs that partially use centralized servers or proactively prefetch relevant content may help counteract the limitations of current broadband connections.

Our results are based on 10 measurement points from 10 different residential broadband networks, and are therefore hardly representative of the Internet at large. However, although small in number, the households in our deployment are quite diverse in terms of Internet access capacity and usage patterns. As future work, it would be interesting to extend our testbed to a larger number of households and ISPs, because a better understanding of broadband traffic will enhance our ability to evaluate different design alternatives. We are also interested in exploring intelligent ways to deliver UGC (e.g., hybrid system, push-based system) [CL09], especially in the presence of content that is popular or large in volume.

5.8. Stratus: a prototype system for sharing UGC from the home

We now quickly describe the high-level design and current implementation of Stratus, a prototype system for home-based UGC sharing. Stratus delivers UGC from standard home gateways and is intended as a proof-of-concept that demonstrates the feasibility of UGC sharing from the home.

5.8.1. Design

Stratus uses home gateways to deliver personal UGC directly from the home. In light of the results of Section 5.6, Stratus serves content from always-on home gateways to achieve good availability. The content shared by users is stored on commodity storage devices attached to the gateway. Software running on the gateway is responsible for delivering the content and mediating access control. In order to grant or deny access to content, requesting clients need to be associated with a user identity. To facilitate the deployment of Stratus and make it easy to use, we decided to use Facebook to authenticate users and perform access control. Authenticating users through Facebook allows Stratus users to immediately share content with their Facebook friends and define flexible access control policies using the same abstractions they use today in Facebook, like for example friend lists. If using a centralized system for authentication is not desirable, Stratus can easily be extended to use self-managed identities that users create and exchange independently of any centralized system, at the cost of a higher deployment bar and lower usability.

5.8.2. Implementation

Our current Stratus implementation runs on NetGearWGT634U home gateways, the same gateways used for the study in Section 5.6, and shown in Figure 5.5. These gateways run OpenWRT [Ope], a Linux distribution for embedded devices. OpenWRT comes with a C++ cross compiler that we used to compile native code for the gateway's MIPS architecture.

Stratus's core functionality is implemented by a personal server (about 8.5K lines of C++ code). The personal server runs on the gateway and performs access control by granting or denying access to requests for the user's content. The gateway also runs two more processes that are used in Stratus: lighttpd [lig], a lightweight web browser, and Samba [sam], an open source implementation of a NAS server used to share files between machines located in the same LAN through the SMB protocol, which is widely supported by all major operating systems. The lightweight web browser delivers the content through the Internet while the Samba server allows users in the local area network to manage the shared content. The shared content resides on a storage device attached to the gateway through a USB port. A special folder on the storage device is used as root directory for all the content shared on Stratus. Users can organize this folder into subfolders and upload data to it from their local home network through the Samba server.

Integration with Facebook

Stratus is integrated with Facebook and uses it to authenticate users. In order to access content shared through Stratus, users have to log on to Facebook and visit the Stratus application canvas page hosted by the Facebook servers. In general, a Facebook application works by forwarding HTTP requests for its canvas page (hosted by Facebook) to an appropriate application server controlled by the application developer.

In the case of Stratus, the application server is the Stratus Facebook application server shown in Figure 5.11. The Stratus application server processes requests coming from Facebook. While processing a request, the application server may interact with a user's personal server to request access to content on behalf of the current user. When the response is ready, the application server sends it back to Facebook, which in turn processes it before sending it to the original user client.

The interaction between Stratus and Facebook is illustrated in Figure 5.11. Communication between Facebook and the Stratus Facebook application server is authenticated through a shared secret assigned by Facebook to the application developer. Similarly, communication between the Stratus application server and each personal server is authenticated using another pairwise shared secret. User identities (e.g. Facebook IDs) are sent as part of each request and response, both between Facebook and Stratus servers as well as between Stratus and personal servers. Personal servers can therefore match the Facebook IDs against their access control lists to grant or deny access to the content.

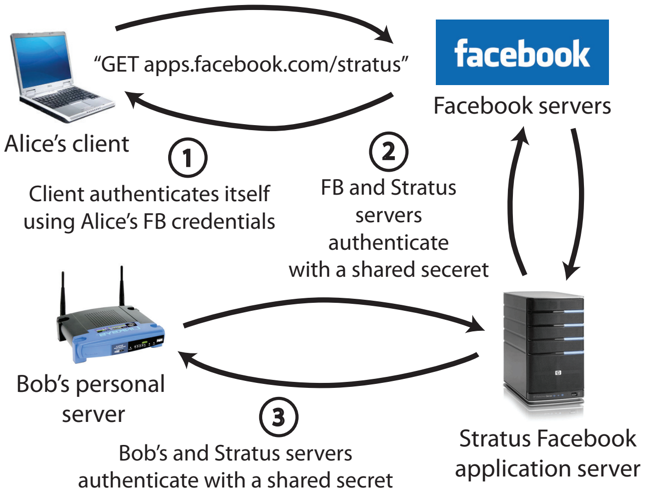

"GET apps.facebook.com/stratus"
facebook
Facebook servers
Alice's client
1
Client authenticates itself using Alice's FB credentials
2
FB and Stratus servers authenticate with a shared seceret
Bob's personal server
3
Bob's and Stratus servers authenticate with a shared secret
Stratus Facebook application server

Figure 5.11: How Stratus uses Facebook for authentication: A user (e.g. Alice) accesses the Stratus Facebook application by visiting a Facebook URL. Facebook forwards all requests to the Stratus application server, which in turn may contact another user's (e.g. Bob's) personal server before sending back a response.

5.8.3. Deployment

We deployed Stratus in the households that participated in the study of Section 5.6. We performed some additional experiments to measure the time taken by Stratus to perform its main operations. Table 5.6 reports the results for a typical setting, which consists of a personal server located behind a residential connection and a client running on a laptop in a remote network.

In order to assess the maximum speed at which our Stratus implementation can deliver content, we performed some experiments where both client and personal server are directly connected through a 100 Mbps Ethernet network. We report the average bandwidth attained when fetching a 50 MB file stored on a USB mass storage device attached to the home gateway. The storage device is either a 4 GB USB flash drive or a 1 TB USB hard disk. The file is fetched through the lighttpd web server and the Samba server. Table 5.7 reports the results. The lightweight web server running on the gateway can achieve a considerable bandwidth (over 10 Mbps). When the content is fetched through the web server from a hard disk, as opposed to a flash drive, the

Operation	Time
Directory listing (19 photos)	3.5 sec
View a photo (low-quality)	3.2 sec
Download of a high-quality photo	14 sec

Table 5.6: End-to-end time required to perform the main Stratus operations: The client runs on a 2.4Ghz Intel Core 2 Duo laptop, the personal server is connected to a home cable connection (450 Kbps upstream). Each value is an average over 10 runs.

	USB mass storage	
Server	USB Flash	USB disk
lighttpd	10.1 Mbps	9.76 Mbps
Samba	13 Mbps	13 Mbps

Table 5.7: Average bandwidth for a 50 MB file transfer from a home gateway to a locally-connected host: Client and personal server are connected through a 100 Mbps Ethernet network.

performance drops by 4%. The Samba server achieves a higher bandwidth (13 Mbps) irrespective of where the content is stored (flash drive or hard disk). These results show that the current implementation of our Stratus server can serve data at a speed that is enough to make efficient use of most Internet upstream connections, whose capacity is currently on the order of a few Mbps or less.

5.9. Discussion

Current trends bode well for adoption of home-based UGC delivery. The bandwidth available to residential Internet connections is rising, with multi-megabit upstream links becoming increasingly common. Prices of commodity storage devices are dropping, and it's not uncommon even for lay users to own multi-terabyte disks that are used to backup personal content, like photos or movies. At the same time, home-gateway devices are becoming more sophisticated, with modern devices equipped with USB ports and significant computing power, often running simple file servers that can turn the device into network-attached storage. Some of these devices run web or FTP servers that can be used to share content with anyone on the Internet [Netb].

In order to share content on the Internet, existing solutions typically require the publisher to send a link to each persons who is supposed to view the content. As a result, it is quite impractical to use these solutions to share content with more than a few users. Moreover, many residential ISPs are known to periodically reassign IP addresses of home connections, thus requiring an external service (such as Dynamic DNS) to keep track of the home server's current IP address. Another problem with existing home-based content sharing solutions is that access control is implemented through a simple shared password. This approach has the obvious inconvenience of requiring the publisher to send the password to each intended viewer along with the link to the content. Also, once the password has been distributed, it's very inconvenient for the publisher to restrict access to the content, because restricting access requires invalidating the old password, creating a new password and distributing it to the new set of intended viewers. In light of these shortcomings, new ways of finding and managing access to user-generated content are required in order to make home-based sharing practical. Stratus is an example of a system that overcomes these shortcomings by integrating home-based UGC sharing with the flexible access control provided by online social networks.

5.9.1. Deployment paths for home-based sharing

Home-gateway devices sophisticated enough to support home-based sharing are becoming a reality. Even though their cost is relatively low when compared to other computing devices, it is still high enough to constitute a hurdle towards wide adoption of home-based content sharing. A possible solution to the cost problem is to have ISPs subsidize these devices. This is already partially happening today, with ISPs providing Internet subscribers with simple wireless routers or network modems used to access the Internet.

In order for ISPs to provide subscribers with more advanced devices, ISPs would need the necessary economic incentives. These incentives could come in different forms. There are existing projects [Nan] that aim to turn residential gateways into a distributed cloud computing infrastructure. The rationale is that ISP-managed home-gateways combine the advantages of cloud computing (centralized control and management) with the advantages of peer-to-peer systems (geographical distribution, proximity to end users). In addition, because home-gateways tend to be on most of the time, they have lower churn rates than traditional peer-to-peer systems. Once ISPs deploy sophisticated home-gateways in subscribers' home and rent them out as a cloud computing resource, they are essentially making profit by partially using subscribers' electricity and bandwidth. Therefore, in order to compensate subscribers, ISPs could let them use the gateways to run useful personal applications, like for example home-based content sharing.

Alternatively, residential ISPs could be interested in having subscribers share their content from home in order to reduce their network costs. This is because sharing content from home is likely to keep a considerable fraction of the existing user-generated traffic within the same geographical region or network, thus reducing the amount of billed traffic that residential ISPs need to send through transit ISPs. Depending on the exact savings, cutting network costs might be a sufficient incentive for residential ISPs to support home-based content sharing.

The second crucial aspect of the deployment of home-based sharing solutions is software. Unlike centralized OSN datacenters that can rely on a team of experts to manage software updates and security, decentralized home-managed solutions are run by lay users. Therefore, there is a concern that personal content-sharing servers might be open to vulnerabilities and security problems, which may result in privacy breaches. A potential way to provide high quality and secure content-sharing software is to leverage the open-source model. Most home-gateway devices already support Linux and can run existing Linux distributions [Ope] tailored for embedded devices. It is therefore conceivable for these home-gateways to run content-sharing servers developed as part of open-source projects. Apart from being free, this software would be developed and maintained by non-profit communities with little or no incentive to abuse users' privacy to pursue commercial profit, which on the other hand is a concern with existing centralized, profit-making OSNs. To provide additional security guarantees, personal servers could run in a virtualized environment and thus be isolated from other software running on the gateway. If the home-gateway is deployed as part of a distributed cloud computing infrastructure owned and managed by ISPs, such a virtualization mechanism is likely to be already in place, as it is the case in traditional cloud-computing solutions.

6. Conclusions

In this section, we summarize the main contributions of this thesis and outline directions for future work.

6.1. Summary

In this thesis, we addressed some implications that emerging Internet workloads have for system design. In particular, we focused on two important emerging workloads: bulk and user-generated content (UGC).

In the context of bulk content, we analyzed the implications of the growing fraction of Internet bulk traffic for ISPs' traffic management. We found that transit ISPs can rely on the delay-tolerant nature of bulk content to deliver it opportunistically using only spare capacity in their largely overprovisioned backbones. We described the design of NetEx, a system that transit ISPs can deploy without fundamental changes to their infrastructure. NetEx allows transit ISPs to deliver considerably more bulk content than what is being delivered today, while using only spare capacity and without penalizing existing traffic. We evaluated NetEx using data from a real Tier-1 ISP spanning three continents. Our evaluation shows that NetEx has the potential to increase the data delivered in the network by 60% - 180% depending on the network topology.

We also explored the opportunities for access ISPs to reduce their peak network traffic by rate limiting only bulk traffic. The rationale is that bulk traffic accounts for a large fraction of the total traffic, and is at the same time more amenable to traffic shaping because it can tolerate longer delays than interactive traffic. Moreover, since access ISPs are often charged by near-peak utilization (e.g. 95^{th} percentile utilization), reducing peak network traffic translates into cost reductions. We systematically analyzed different traffic shaping policies, and found a policy that greatly reduces peak network traffic (by about 50%) while at the same time minimizing the negative impact on traffic shaped flows. However, we also found that multiple traffic shapers on the same network path result in significant performance losses for bulk flows in transit. To overcome this problem, we proposed in-network staging of bulk flows as a way to temporarily buffer the data of bulk flows, thus avoiding the negative interaction of multiple traffic shapers.

With respect to user-generated content, we discussed the feasibility of a system that enables users to share UGC directly from home, as opposed to using a centralized OSN website. Delivering content from home can potentially free users from the limitations, privacy policies and terms of use imposed by OSN providers. Using data from two popular OSN sites, and a study of the availability and performance of a number of home network connections, we estimated the fraction of existing OSN users who would be able to share their content directly from their home networks. We found that, to reach sufficient availability, content needs to be shared from always-on, always-connected devices like home gateways, as opposed to desktop or laptop computers. We also found that, depending on their network resources, a considerable fraction of users would not be able to share all their UGC from home. The main reason for this is that some UGC occasionally reaches high popularity and requires well-provisioned servers to be shared. At the same time, a lot of UGC is unlikely to ever reach such high popularity, either because it is of a personal nature or protected by privacy restrictions. Therefore, current trends bode well for the delivery of personal UGC from home. This is useful because sharing personal content from home can potentially give users better privacy guarantees and protect them against privacy infringements from OSN providers. Therefore, we proposed a hybrid system where users only share personal content from home, but continue to share highly popular objects from traditional OSN websites.

Finally, we presented Stratus, a system that allows users to share their private content from home. Stratus uses home gateways with attached USB mass storage as servers. Users can access Stratus through an existing OSN (e.g. Facebook), and can thus employ already-established identities and social links. Alternatively, if stricter privacy guarantees are needed, Stratus can easily be extended to use self-managed identities that completely free users from OSN providers. The downside is that users would have to create and exchange their own Stratus identities in order to establish trusted social links.

6.2. Directions for future research

With respect to bulk data delivery, an interesting direction for future work is quantifying the extent to which spare capacity can be used to deliver video-on-demand (VOD) content. Such content has recently gained high popularity thanks to very popular video-sharing services like YouTube. This content is likely to account for a considerable fraction of Internet bulk traffic, yet it has more stringent delivery requirements than other bulk traffic like, say, downloads happening in the background, which don't require user interaction. In particular, it would be interesting to investigate how traffic shaping and opportunistic delivery techniques that target bulk traffic need to be modified to account for VOD content. Since traffic shaping will necessarily have to be less aggressive when dealing with VOD traffic, it would be interesting to explore the tradeoff between the effectiveness of traffic shaping and its impact on VOD performance. However, in order to perform such analysis, more detailed traces from the commercial Internet are required. In particular, traces containing application-level information are needed to identify VOD flows and quantify their prevalence and characteristics (such as bandwidth demands) in the current Internet. Obtaining such detailed traces from commercial ISPs is notoriously difficult for privacy and business reasons. Detailed traces from academic networks are easier to obtain; however, it is harder to directly apply findings derived from academic traces to the commercial Internet.

With respect to sharing UGC from home, we see Stratus as a good starting point towards freeing users from the constraints and limitations of centralized personal data sharing. Because the performance of the current Stratus design is ultimately limited by the capacity of residential Internet connections, it may be hard to effectively deliver high-quality media content (like good quality video) directly from the Stratus personal servers. In these cases, it makes sense to explore hybrid designs in which a Stratus personal server is assisted by other personal servers, or even a trusted centralized service, when delivering high-bandwidth content. Such a hybrid system exposes an interesting tradeoff between effectiveness of content delivery and control of the shared data. While other personal servers that help deliver the content can increase the performance, they may also reduce the control that users have of their personal data. A possible way to alleviate this problem is to let only users who have recently accessed the content, and have thus received access permission from the publisher, help in the delivery of the content. It is conceivable that, if the penetration of the system is large enough, there will be at least some users who have accessed the content in the recent past and were able to cache it at their personal servers. A larger-scale deployment of a system like Stratus could help determine how much content shared in practice requires such a collaborative delivery, and how the collaborative system could be designed in order to increase the performance of content delivery while at the same time maximizing the control users have of their personal content.

Bibliography

[ABI] Abilene Backbone Network. http://abilene.internet2.edu/.

[AG03] Nadia Ben Azzouna and Fabrice Guillemin. Analysis of ADSL Traffic on an IP Backbone Link. In *Proceedings of the IEEE Global Telecommunications Conference*, 2003.

[Aka] http://www.akamai.com.

[AMA] Amazon Simple Storage Service. http://aws.amazon.com/s3/.

[AMD09] Demetris Antoniades, Evangelos P. Markatos, and Constantine Dovrolis. One-Click Hosting Services: A File-Sharing Hideout. In *Proceedings of the ACM Internet Measurement Conference*, 2009.

[ANB05] Sharad Agarwal, Antonio Nucci, and Supratik Bhattacharrya. Measuring the Shared Fate of IGP Engineering and Interdomain Traffic. In *Proceedings of the IEEE International Conference on Network Protocols*, 2005.

[Apa] Apache HTTP Server. http://httpd.apache.org/.

[BB95] A. Bakre and B. R. Badrinath. I-TCP: Indirect TCP for Mobile Hosts. In *Proceedings of the International Conference on Distributed Computing Systems*, 1995.

[BBC$^+$98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Service. RFC 2475 (Informational), December 1998. http://www.ietf.org/rfc/rfc2475.txt.

[BDJT02] Supratik Bhattacharrya, Christophe Diot, Jorjeta Jetcheva, and Nina Taft. Geographical and Temporal Characteristics of Inter-POP Flows: View from a Single POP. *European Transactions on Telecommunications*, February 2002.

[Bel] Comments of Bell Aliant Regional Communications, Limited Partnership and Bell Canada. http://www.crtc.gc.ca/PartVII/eng/2008/8646/c12_200815400.htm#2b.

[BIT] BitTorrent Homepage. www.bittorrent.org.

[Bkc02] Nevil Brownlee and kc claffy. Understanding Internet Traffic Streams: Dragonflies and Tortoises. *IEEE Communications Magazine*, Oct 2002.

[BKea01] J. Border, M. Kojo, and J. Griner et al. Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. RFC 3135, 2001. http://www.faqs.org/rfcs/rfc3135.html.

[BPea02] H. Balakrishnan, V. N. Padmanabhan, and G. Fairhurst et al. TCP Performance Implications of Network Path Asymmetry. RFC 3449, 2002. http://www.faqs.org/rfcs/rfc3449.html.

[BRCA09] Fabricio Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgilio Almeida. Characterizing User Behavior in Online Social Networks. In *Proceedings of the ACM Internet Measurement Conference*, 2009.

[BS97] Kevin Brown and Suresh Singh. M-TCP: TCP for Mobile Cellular Networks. *SIGCOMM Computer Communication Review*, 27(5):19–43, 1997.

[BSVD09] Sonja Buchegger, Doris Schiöberg, Le Hung Vu, and Anwitaman Datta. PeerSoN: P2P Social Networking - Early Experiences and Insights. In *Proceedings of the ACM Social Network System Workshop*, 2009.

[BU97] Azer Bestavros and Ibrahim Matta Boston University. Load Profiling for Efficient Route Selection in Multi-Class Networks. In *Proceedings of the IEEE International Conference on Network Protocols*, 1997.

[CFEK06] Kenjiro Cho, Kensuke Fukuda, Hiroshi Esaki, and Akira Kato. The Impact and Implications of the Growth in Residential User-to-user Traffic. In *Proceedings of the ACM Annual Conference of the Special Interest Group on Data Communication*, 2006.

[CFSD90] J. Case, M. Fedor, M. Schoffstall, and J. Davin. A Simple Network Management Protocol (SNMP). RFC 1157, 1990. http://tools.ietf.org/rfc/rfc1157.txt.

[Cisa] Cisco Systems. Cisco QoS. http://www.cisco.com/en/US/products/ps6558/products_ios_technology_home.html.

[Cisb] Cisco IOS Classification. http://www.cisco.com/en/US/docs/ios/12_2/qos/configuration/guide/qcfclass.html.

[Cisc] Cisco IOS Congestion Management. http://www.cisco.com/en/US/docs/ios/12_2/qos/configuration/guide/qcfconmg_ps1835_TSD_Products_Configuration_Guide_Chapter.html.

[CKR$^+$07] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn, and Sue Moon. I Tube, You Tube, Everybody Tubes: Analyzing the World's Largest User Generated Content Video System. In *Proceedings of the ACM Internet Measurement Conference*, 2007.

[CL09] Xu Cheng and Jiangchuan Liu. NetTube: Exploring Social Networks for Peer-to-Peer Short Video Sharing. In *Proceedings of the Annual IEEE International Conference on Computer Communications*, 2009.

[CMG09] Meeyoung Cha, Alan Mislove, and Krishna P. Gummadi. A Measurement-driven Analysis of Information Propagation in the Flickr Social Network. In *Proceedings of the International World Wide Web Conference*, 2009.

[Com] Comcast: Description of planned network management practices. `http://downloads.comcast.net/docs/Attachment_B_Future_Practices.pdf`.

[cpl] ILOG CPLEX. `http://www.ilog.com`.

[DHGS07] Marcel Dischinger, Andreas Haeberlen, Krishna P. Gummadi, and Stefan Saroiu. Characterizing Residential Broadband Networks. In *Proceedings of the ACM Internet Measurement Conference*, 2007.

[Diaa] Diaspora OSN. `http://www.joindiaspora.com`.

[diab] Jim Dwyer, Four nerds and a cry to arms against Facebook. NYTimes, 2010. `http://tinyurl.com/25a2o8t`.

[DMHG08] Marcel Dischinger, Alan Mislove, Andreas Haeberlen, and Krishna P. Gummadi. Detecting BitTorrent Blocking. In *Proceedings of the ACM Internet Measurement Conference*, October 2008.

[ELE] Average retail price of electricity, U.S. Energy Information Administration, 2010. `http://tinyurl.com/525u28`.

[Elm08] Philip Elmer-DeWitt. iTunes store: 5 billion songs; 50,000 movies per day, June 19th, 2008. `http://tech.fortune.cnn.com/2008/06/19/itunes-store-5-billion-songs-50000-movies-per-day`.

[emua] eMule Project. `http://www.emule-project.net`.

[EMUb] Emulab - network emulation testbed. `http://www.emulab.net`.

[faca] Nick Bilton, Price of Facebook privacy? start clicking. NYTimes, 2010. `http://tinyurl.com/39nyzfb`.

[FACb] Facebook Statistics. `http://www.facebook.com/press/info.php?statistics`.

[FACc] Facebook terms of use. `http://www.facebook.com/#!/terms.php?ref=pf`.

[Fac09] Needle in a Haystack: Efficient Storage of Billions of Photos, 2009. Facebook Engineering Notes, `http://tinyurl.com/cju2og`.

[Fas] FastTrack P2P network. `http://developer.berlios.de/projects/gift-fasttrack/`.

[FFM04] Michael J. Freedman, Eric Freudenthal, and David Mazières. Democratizing Content Publication with Coral. In *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*, 2004.

[FGea99] R. Fielding, J. Gettys, and J. Mogul et al. Hypertext Transfer Protocol – HTTP/1.1. RFC 2616, 1999. `http://www.faqs.org/rfcs/rfc2616.html`.

[Flo91] Sally Floyd. Connections with Multiple Congested Gateways in Packet-switched Networks Part 1: One-way Traffic. *ACM Computer Communication Review*, 21:30–47, 1991.

[Fri02] Steven Friederich. Bandwidth restrictions save almost $1 million, Oct 22nd, 2002. `http://thedaily.washington.edu/2002/10/22/bandwidth-restrictions-save-almost-1-million/`.

[FT00] Bernard Fortz and Mikkel Thorup. Internet traffic engineering by optimizing OSPF weights. In *Proceedings of the Annual IEEE International Conference on Computer Communications*, 2000.

[GALM07] Phillipa Gill, Martin Arlitt, Zongpeng Li, and Anirban Mahanti. YouTube Traffic Characterization: A View From the Edge. In *Proceedings of the ACM Internet Measurement Conference*, 2007.

[GJT04] Anders Gunnar, Mikael Johansson, and Thomas Telkamp. Traffic Matrix Estimation on a Large IP Backbone - A Comparison on Real Data. In *Proceedings of the ACM Internet Measurement Conference*, 2004.

[GK06] Eric Gourdin and Olivier Klopfenstein. Comparison of Different QoS-oriented Objectives for Multicommodity Flow Routing Optimization. In *Proceedings of the IEEE International Conference on Telecommunications*, 2006.

[GM02] Liang Guo and Ibrahim Mitta. Scheduling Flows with Unknown Sizes: An Approximate Analysis. In *Proceedings of ACM SIGMETRICS*, 2002.

[Gnu] Gnutella P2P network. `http://rakjar.de/gnufu/index.php/Main_Page`.

[GQX+04] David K. Goldenberg, Lili Qiu, Haiyong Xie, Yang Richard Yang, and Yin Zhang. Optimizing Cost and Performance for Multihoming. In *Proceedings of the ACM Annual Conference of the Special Interest Group on Data Communication*, 2004.

[GSG02] Krishna P. Gummadi, Stefan Saroiu, and Steven D. Gribble. King: Estimating Latency between Arbitrary Internet End Hosts. In *Proceedings of the ACM Internet Measurement Workshop*, Marseille, France, 2002.

[HK99] Thomas R. Henderson and Randy H. Katz. Transport Protocols for Internet-Compatible Satellite Networks. *IEEE Journal on Selected Areas in Communications*, 17:326–344, 1999.

[HR08] Jiayue He and Jennifer Rexford. Towards Internet-wide Multipath Routing. *IEEE Network Magazine*, 2008.

[IBTD03] Sundar Iyer, Supratik Bhattacharrya, Nina Taft, and Christophe Diot. An Approach to Alleviate Link Overload as Observed on an IP Backbone. In *Proceedings of the Annual IEEE International Conference on Computer Communications*, San Francisco, March 2003.

[ipo] Ipoque Internet Study 2008/2009. `http://www.ipoque.com/userfiles/file/ipoque-Internet-Study-08-09.pdf`.

[Ipo09] Ipoque GmbH. OpenDPI, 2009. `http://www.opendpi.org`.

[Kar02] G. Karakostas. Faster Approximation Schemes for Fractional Multicommodity Flow Problems. In *Proceedings of ACM/SIAM Symposium on Discrete Algorithms*, 2002.

[KBB$^+$04] Thomas Karagiannis, Andre Broido, Nevil Brownlee, KC Claffy, and Michalis Faloutsos. Is P2P Dying or just Hiding? In *Proceedings of the IEEE Global Communications Conference*, 2004.

[KBFc04] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and Kc claffy. Transport Layer Identification of P2P Traffic. In *Proceedings of the ACM Internet Measurement Conference*, 2004.

[KKFT02] S. Kopparty, S. V. Krishnamurthy, M. Faloutsos, and S. K. Tripathi. Split TCP for Mobile Ad-hoc Networks. In *Proceedings of the IEEE Global Communications Conference*, 2002.

[KLM97] Tom M. Kroeger, Darrell D. E. Long, and Jeffrey C. Mogul. Exploring the Bounds of Web Latency Reduction from Caching and Prefetching. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, 1997.

[KMC$^+$00] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The Click Modular Router. *ACM Transactions on Computer Systems*, 18(3):263–297, 2000.

[LB08] Matthew M. Lucas and Nikita Borisov. FlyByNight: mitigating the privacy risks of social networking. In *Proceedings of the ACM Workshop on Privacy in the Electronic Society*, 2008.

[Lei09] Tom Leighton. Improving Performance on the Internet. *Communications of the ACM*, 2009.

[lig] Lighttpd server. `http://www.lighttpd.net/`.

[Lin] Linux Advanced Routing & Traffic Control HOWTO. `http://lartc.org/lartc.html`.

[LPC+04] Anukool Lakhina, Konstantina Papagiannaki, Mark Crovella, Christophe Diot, Eric D. Kolaczyk, and Nina Taft. Structural Analysis of Network Traffic Flows. In *Proceedings of ACM SIGMETRICS*, 2004.

[LPP04] K. Lakshminarayanan, V. Padmanabhan, and J. Padhye. Bandwidth Estimation in Broadband Access Networks. In *Proceedings of the ACM Internet Measurement Conference*, 2004.

[LSRS09] N. Laoutaris, G. Smaragdakis, P. Rodriguez, and R. Sundaram. Delay Tolerant Bulk Data Transfers on the Internet. In *Proceedings of SIGMETRICS*, 2009.

[MDea00] G. Montenegro, S. Dawkins, and M. Kojo et al. Long Thin Networks. RFC 2757, 2000. `http://tools.ietf.org/html/rfc2757`.

[MFPA09] Gregor Maier, Anja Feldmann, Vern Paxson, and Mark Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *Proceedings of the ACM Internet Measurement Conference*, 2009.

[MG00] Ibrahim Matta and Liang Guo. Differentiated Predictive Fair Service for TCP Flows. In *Proceedings of the IEEE International Conference on Network Protocols*, 2000.

[Nan] NanoDataCenters Project. `http://www.nanodatacenters.eu/`.

[Neta] Netflix Online Movie Rental. `http://www.netflix.com`.

[Netb] Netgear WNDR3700. `http://www.netgear.com/ultimatewifi`.

[Netc] Netgear WGT634U. `http://nuwiki.openwrt.org/oldwiki/OpenWrtDocs/Hardware/Netgear/WGT634U`.

[Nie] Nielsen Online Report. Social Networks & Blogs Now 4th Most Popular Online Activity, 2009. `http://tinyurl.com/cfzjlt`.

[NTBT04] Antonio Nucci, Nina Taft, Chadi Barakat, and Patrick Thiran. Controlled Use of Excess Backbone Bandwidth for Providing New Services in IP-over-WDM Networks. *IEEE Journal on Selected Areas in Communications - Optical Communications and Networking series*, November 2004.

[Ope] OpenWrt. `www.openwrt.org`.

[paca] Packeteer. `http://www.packeteer.com`.

[pacb] Blue Coat PacketShaper. `http://bluecoat.com/products/packetshaper`.

[Pog] Pogoplug. `http://www.pogoplug.com`.

[PPPW05] Rong Pan, Balaji Prabhakar, Konstantinos Psounis, and Damon Wischik. SHRiNK: A Method for Enabling Scalable Performance Prediction and Efficient Network Simulation. *IEEE Transactions on Networking*, 13, October 2005.

[rfc98] Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474, 1998. `http://tools.ietf.org/html/rfc2474`.

[Röt08] Janko Röttgers. Internetanbieter bremst Tauschbörsen aus. Focus Online, Mar 6th, 2008. `http://www.focus.de/digital/internet/kabel-deutschland_aid_264070.html`.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. RFC 3031, January 2001. `http://www.ietf.org/rfc/rfc3031.txt`.

[S.] Metha S. Verizon's big bet on fiber optics. `http://money.cnn.com/magazines/fortune/fortune_archive/2007/03/05/8401289`.

[sam] Samba. `http://www.samba.org/`.

[SM90] Farhad Shahrokhi and D. W. Matula. The Maximum Concurrent Flow Problem. *Journal of the ACM*, 1990.

[Spa] T. Spangler. ATT: U-verse tv spending to increase. `http://www.multichannel.com/article/ca6440129.html`.

[Spra] Sprint AR&ATL. `http://research.sprintlabs.com`.

[Sprb] Sprint Internet Access SLA. `http://www.sprint.com/business/resources/dedicated_internet_access.pdf`.

[Squ] Squid proxy cache. `http://www.squid-cache.org`.

[SRS99] Anees Shaikh, Jennifer Rexford, and Kang G. Shin. Load-Sensitive Routing of Long-Lived IP Flows. In *Proceedings of the ACM Annual Conference of the Special Interest Group on Data Communication*, 1999.

[ST01a] S. Shalunov and B. Teitelbaum. Qbone Scavenger Service (QBSS) Definition. Technical report, March 2001. `http://qos.internet2.edu/wg/wg-documents/qbss-definition.txt`.

[ST01b] Stanislav Shalunov and Benjamin Teitelbaum. TCP Use and Performance on Internet2. In *Proceedings of the ACM Internet Measurement Workshop*, 2001.

[SVCC09] Amre Shakimov, Alexander Varshavsky, Landon P. Cox, and Ramón Cáceres. Privacy, cost, and availability tradeoffs in decentralized OSNs. In *Proceedings of the USENIX Workshop on Online Social Networks*, 2009.

[SWB$^+$03] Subhash Suri, Marcel Waldvogel, Daniel Bauer, , and Priyank Ramesh Warkhede. Profile-Based Routing and Traffic Engineering. *Computer Communications*, March 2003.

[SZF10] Jinyuan Sun, Xiaoyan Zhu, and Yuguang Fang. A Privacy-Preserving Scheme for Online Social Networks with Efficient Revocation. In *Proceedings of the Annual IEEE International Conference on Computer Communications*, 2010.

[TMSV03] R. Teixeira, K Marzullo, S Savage, and G. Voelker. Characterizing and Measuring Path Diversity in Internet Topologies. In *Proceedings of ACM SIGMETRICS*, June 2003.

[Top07] Robb Topolski. Comcast is using Sandvine to manage P2P connections, May 2007. `http://www.dslreports.com/forum/r18323368-Comcast-is-using-Sandvine-to-manage-P2P-Connections`.

[twi] Twitter Facts and Figures. `http://www.viralblog.com/research/twitter-facts-figures/`.

[Uni07] Universities Prepare for Data Deluge from CERN Collider, May 2007. `http://www.hpcwire.com/hpc/1572567.html`.

[Us] US broadband's average speed, 2010. `http://tinyurl.com/yamsagn`.

[VKD02] A. Venkataramani, R. Kokku, and M. Dahlin. TCP-Nice: A Mechanism for Background Transfers. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation*, December 2002.

[Youa] YouTube Fact Sheet. `http://www.youtube.com/t/fact_sheet`.

[Youb] YouTube most popular videos. `http://www.readwriteweb.com/archives/top_10_youtube_videos_of_all_time.php`.

[ZDA06] Yong Zhu, Constantinos Dovrolis, and Mostafa Ammar. Dynamic Overlay Routing Based on Available Bandwidth Estimation. *Computer Networks Journal*, 2006.