WIDE-AREA ROUTE CONTROL FOR ONLINE SERVICES

A Thesis
Presented to
The Academic Faculty

by

Vytautas Valancius

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
May 2012

WIDE-AREA ROUTE CONTROL FOR ONLINE SERVICES

Approved by:

Dr. Nick Feamster, Advisor
College of Computing
Georgia Institute of Technology

Dr. Vijay V. Vazirani
College of Computing
Georgia Institute of Technology

Dr. Mostafa Ammar
College of Computing
Georgia Institute of Technology

Dr. Ramesh Johari
Computer Science
Stanford University

Dr. Ellen Zegura
College of Computing
Georgia Institute of Technology

Date Approved: January 19th, 2012

ACKNOWLEDGEMENTS

One winter night, Nick Feamster, my advisor, and I were chatting about outdoor activities we like. I mentioned that in Lithuania, my home country, cross-country skiing is quite popular. ”We should do it here in Atlanta!” Nick said. ”How would we do that in the South?” I replied, ”We’ll never get enough snow!” Guess what? Two weeks later Atlanta got just enough snow for us to ski in Piedmont Park. This snippet illustrates well Nick’s attitude - he is a shameless optimist and he often believed in me more than I believed in myself. He is also a hard worker, who would go for 10 mile-plus jogs after a midnight deadline, while on a call with me discussing my future research directions. His bright outlook is what primarily sustained me throughout my graduate studies; his work ethic is what inspired me to become even better. I think I could write the whole dissertation how awesome Nick is, but I think I’ll just stop here and say that I had the best advisor one could ever have.

I was also lucky to work with professor Jennifer Rexford, with whom I coauthored one of my publications. Jennifer is a very warm person and I always appreciated her advice on both my academic and my personal development. I also had a lot of fun to work with professor Ramesh Johari. When I would doubt my research, he’d say that I’m having my ”Oprah moment.” Ramesh was also indispensable in shaping my work on tiered pricing, guiding me through the field of economics and econometrics. Credit also goes to my thesis committee members Mostafa Ammar, Vijay V. Vazirani and Ellen Zegura, who helped to shape my work. I want to also thank professor Akihiro Nakao who helped me with Transit Portal work and with that last bottle of sake in a bar close to the Hakusan subway station in Tokyo. During my internship in Technicolor, Paris, I had a lot of fun jogging in Parc de Saint-Cloud with my hosts Christophe Diot and Laurent Massoulié. Christophe would

always make sure he finishes first, while Laurent would just enjoy the jog. I want to acknowledge Pablo Rodriquez and Nikolaos Laotaris who made my internship in Telefonica Research, Barcelona possible. I want to also thank my internship hosts at Google: Puneet Sood, Rui Zhang-Shen, and Anand Kanagala, who were instrumental in shaping my career path. Rui wasn't even my official mentor and yet she was always there for me when I needed guidance. I would also like to acknowledge Andy Bavier, Martin Casado, David Clark, Edward Crabbe, Maurice Dean, Vijay Gill, Sharon Goldberg, Ramesh Govindan, and David Maltz, with whom I had many fruitful discussions and who, sometimes without knowing it, influenced my research by a great deal.

Graduate studies can be a taxing endeavor. My friends, in the lab and outside it, kept me sane through this journey. I want to thank my friend, roommate, labmate, and, now, colleague Murtaza Motiwala for all the late night discussions we had about the life, the universe, and the number 42. (Murtaza, you still haven't convinced me that human population can grow indefinitely!) In the lab, I also had a lot of fun moments with Hyojoon Kim, who always pulled his hair when excited; Srikanth Sundaresan, who forced everyone to watch Monty Python sketches; Sam Burnett, who was the only American in the lab and who had to assure as that Sarah Palin will not be the next president of the U.S.; Cristian Lumezanu, who's car I shamelessly used when he was away; Robert Lychev, who scared everyone with his Kung-Fu skills; Saamer Akhshabi, who once took 24 beer cans to a day hike; Ilias Foudalis, who knew more than anyone in the lab about the meteorology; Samantha Lo, who sometimes cooked cakes for the whole lab for no reason; Mukarram Bin Tariq, who was always there if I needed help; Yogesh Mundada, who is an amazing cook; Ahmed Mansy, who has suffered too much in the hands of TSA; and Steve Woodrow, who's manners made us feel uncultured.

Outside the lab, I had a large circle of friends who supported me: Ignas Kukenys, a fellow who started his Ph.D. in New Zealand after I started mine and finished before I finised mine; Remigijus Staniulis, who's not a researcher, nor a grad student, and yet crashed a

USENIX ATC conference and enjoyed the talks even more than me; Elijus Civilis, who inspires me to use my bicycle even more; Sapan Bhatia, who recently became a father to two beautiful twins; Wolfgang Mülbauer, who confirmed my stereotype that all Germans are very organized; and Paul Gush, who is perfect at imitating my Russian-sounding accent. I would like to specifically acknowledge Xue Cai, who reads history books and thinks that Cartman is the cutest of the South Park characters.

Finally, I want to thank my mother Leonija, my sister Laima, and her husband Mindaugas, who were always there rooting for me and lifting me up when I was down.

BIBLIOGRAPHIC NOTES

Chapter 2 includes material from a paper published in USENIX ATC 2010 [109] and from a poster in CoNext 2007 [107]. Chapter 4 includes material from a papers published in SIGCOMM 2011 [110] and in ReArch 2008 [108]. The datasets I collected and described in Chapter 3 are publicly available from my website [9].

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Accelerated by on-demand computing, the number and diversity of the Internet services is increasing. Such online services often have unique requirements for the underlying wide-area network: For instance, online gaming service might benefit from low delay and jitter paths to client, while online data backup service might benefit from cheaper paths. Unfortunately, today's Internet does not accommodate fine-grained, service-specific wide-area route control. In this dissertation, I achieve the following goals: 1) *improve the access* to the routes, 2) *quantify the benefits* of fine-grained route control, and 3) *evaluate the efficiency of current payment schemes* for the wide-area routes.

- **Improving access to wide-area route control.** Online services face significant technological and procedural hurdles in *accessing* the routes: Each service in need to control the Internet routes, has to obtain own equipment, Internet numbered resources, and establish contracts with upstream ISPs. In this dissertation, I propose and describe implementation and deployment of a secure and scalable system which provides on-demand access to the Internet routes. In setting such as cloud data center, the system can support multiple online services, providing each service with an illusion of direct connectivity to the neighboring Internet networks, which, for all practical purposes, allows services to participate fully in the Internet routing.
- **Quantifying the benefits of fine-grained route control.** Even if online services are presented with wide-area route choice, it is not clear how much *tangible benefit* such choice provides. Most modern Online Service Providers (OSP) rely primarily on the content routing to improve network performance between the clients and the replicas. In this dissertation, I quantify the potential benefit the OSPs can gain if they perform a joint network and content routing. Among other findings, I find that by performing

joint content and network routing, OSPs can achieve 22% larger latency reduction than can be obtained by content routing alone.

- **Modeling and evaluating the efficiency of the current payment schemes for wide-area routes.** Finally, increasing diversity and sophistication of the online services participating in the Internet routing poses a challenge to *payment* models used in today's Internet. Service providers today charge business customers a blended rate: a single, "average" price for unit of bandwidth, without regard to cost or value of individual customer's flows. In my dissertation, I set to understand how efficient this payment model is and if more granular payment model, accounting for the cost and value of different flows could increase the ISP profit and the consumer surplus. I develop an econometric demand and cost model and map three real-world ISP data sets to it. I find that ISPs can indeed improve the economic efficiency with just a few pricing tiers.

CHAPTER I

INTRODUCTION

The Internet has grown in many ways over the past couple of decades. From early 90's to today, the number of people with an access to the Internet grew from a few thousands to almost two billion [10], while data consumed by these Internet users increased from a few hundreds kilobytes to a hard-to-comprehend exabyte a day [38]. Internet usage patterns also evolved as consumers use the Internet for everyday activities, such as entertainment and banking. The number of different Internet uses is further accelerated by new service delivery technologies, such as "cloud computing" [21, 49, 113], which enables on-demand provisioning of computing and storage resources for new uses. The underlying theme of this change is that Internet, as a platform, carries ever increassing diversity of services, each with unique requirements for network performance and cost. For instance, bandwidth intensive services might prefer cheaper paths in the network, financial online services might favor reliable paths, while interactive online service might value performance the most.

Despite the growing number and diversity of supported online services, the basic wide-area routing technology and the prevailing Internet business models remaine stagnant. The Internet comprises of approximately 30,000 networks of various sizes, who control routing using a reliable yet arcane Border Gateway Protocol (BGP) [95]. The BGP allows participating networks to choose paths the traffic will take. Unfortunately, due to historical and practical reasons, *access* to such wide-area route control is only available to larger networks with significant resources — such as service providers, university campuses, or content providers. New and emerging online services, such as operating from cloud computing platforms, cannot exploit the Internet route diversity. For instance, a cloud computing provider, such as Amazon EC2, can run thousands of hosted services in a data center and

all such hosted services use identical wide-area routes, even if the network requirements of such services varies greatly.

Even if an online service attains access to the Internet route control, it is not at all clear how much benefit it could attain. Services today primarily use replication and content routing to reduce latency and increase throughput. Allowing services to also perform granular network routing is not trivial and thus operators need to assess potential benefits before performing joint network and content routing.

Finally, Internet *business models* are challenged by changing requirements of the online services. Historically, networks in the Internet priced connectivity indiscriminately, without regard to the value the services gain from the network or the cost which individual services incur on the network. While this model worked fine with few types of services, it is unclear if such indiscriminate pricing is optimal with a new reality of multiple online services with diverse demands.

The clash between the growing diversity of the Internet services and the rigidity of the wide-area routing technology and business models outlined above is the underlying theme of this dissertation. Most of the current work in this area deals with online services and Internet route control separately. For example, large body of work focus on enhancing wide-area route control for networks [51, 103, 115], but stops short of explaining how networked services could access such control without fundamental changes to the Internet routing infrastructure. In the other side of the spectrum is the work that explores how modern networked services, such as search engines, video streaming sites, online games, and social networks can adapt the the realities of the Internet [14, 114, 117]. In this dissertation I explore and bridge this gap. At a high-level, I raise the following questions:

1. How can new online services attain access to the Internet route control at a level similar to one enjoyed currently by large ISPs and enterprises?
2. How much benefit a fine-grained network routing adds when used in conjunction with a conventional content routing?

3. Are the current Internet business models adequate to support the increasing diversity of online services?

1.1 Thesis Contributions

In this dissertation I raise the following hypothesis: *Online services can obtain access to and gain benefit from wide-area routing in a scalable and secure manner. Such access can be accomplished using using commodity software and hardware components, and without major changes to the Internet infrastructure or business models*. More specifically, I raise three claims with distinct contributions as presented below:

1. **Route control.** I claim that it is possible to provide, granular wide-area route control to online services in a scalable and secure fashion. In Section 2.2, I detail the requirements for such routing and describe the system that can meet these requirements.

   Specifically, I have implemented a Transit Portal system [12], which consists of five Transit Portal nodes in locations across the United States. The system allows researchers to access full-fledged wide-area route control. The system is currently used by Georgia Tech and University of Washington researchers to conduct inter-domain routing research.

   The original work addressing this hypothesis was published in CoNEXT'07 [107] and USENIX ATC'10 [109] conferences. Primary contribution of this work is showing that it is possible to access full wide-area route control on-demand and that it scales well.
2. **Quantifying performance gains.** I claim that online services can and should employ fine-grained network routing in conjunction with a conventional content routing. In Chapter 3, I show why online services should perform joint network and content routing. Specifically, I collect extensive round-trip-time, bandwidth, jitter, and out-of-order packet delivery measurements between a number of service replicas, hundreds of clients and across hundreds of wide-area paths. The measurement study is

the first of its kind and contains millions of records. Data analysis shows that, for example, OSPs can achieve 22% larger latency reduction than can be obtained by content routing alone.

3. **Route pricing.** I claim that current connectivity pricing models in the Internet are sufficient to provide efficient routing even for a diverse set of online services. In Chapter 4 I describe: 1) current pricing models between the ISPs, 2) outline the models for studying the efficciency of such pricing, and 3) evaluate the effectiveness of current payment system between ISPs.

   Specifically, I find that just by using 2-3 pricing tiers for their transit pricing, ISPs can attain about 80-90% if the gain they would attain if they were to price transit at inifinite granularity. I also find that consumer surplus follows closely the trends set by ISP profit gains.

   The original work addressing this hypothesis was published in ReArch'08 [108] and SIGCOMM'11 [110] conferences. Primary contributions of this work are: 1) a novel way to model Internet bandwidth market, and 2) observation that 2-3 bandwidth pricing tiers can capture 80-90% of profit and consumer surplus obtainable with infinite number of pricing tiers.

1.2 Roadmap

The rest of the thesis is organized as follows:

- Chapter 2 presents the implementation of Transit Portal platform that enables access to fine-grained wide-area routing for researchers and serves as a proof-of-concept that such access is possible for online services as well. The focus in this chapter is on the design and implementation of such system.
- Chapter 3 presents a measurement study of the benefits the fine-grained route control can give to OSPs that only use conventional content routing. In this chapter I collect and analyse an extensive wide-area route performance measurement set.

- Chapter 4 presents models for studying the efficiency of current business relations between the Internet Service Providers and the Online Service Providers. The focus in this chapter is on the modeling Internet transit market and estimating ISP profit and consumer surplus for different transit pricing strategies.
- Chapter 5 presents concluding remarks and lists a few directions for future research.

How to read this thesis? The best way to read this thesis would be to read chapters in serial order as outlined. Another way is to read Chapters 2 and 3 as implementation and evaluation of Transit Portal platform, and then read Chapter 4 separately as a study of overall Internet transit market.

CHAPTER II

CONTROLLING WIDE-AREA ROUTES WITH TRANSIT PORTAL

Cloud-based hosting platforms make computational resources a basic utility that can be expanded and contracted as needed [21, 113]. However, some distributed services need more than just computing and bandwidth resources—they need control over the *network*, particularly over the wide-area routes to and from their users. More flexible route control helps improve performance [17, 18, 37] and reduce operating costs [53]. For example, interactive applications like online gaming want to select low-latency paths to users, even if cheaper or higher-bandwidth paths are available. As another example, a service replicated in multiple locations may want to use IP anycast to receive traffic from clients and adjust where the address block is announced in response to server failures or shifts in load.

Although flexible route control is commonplace for both content providers and transit networks, today's cloud-based services do not enjoy the same level of control over routing. Today, the people offering these kinds of distributed services have two equally unappealing options. On the one hand, they could build their own network footprint, including acquiring address space, negotiating contracts with ISPs, and installing routers. That is, they could essentially become network operators, at great expense and with little ability to expand their footprint on demand. On the other hand, they could contract with a hosting company and settle for whatever "one size fits all" routing decisions this company's routers make.

This missed opportunity is not for a lack of routing diversity at the data centers: for example, RouteViews shows that Amazon's Elastic Cloud Computing (EC2) has at least 58 upstream BGP peers for its Virginia data center and at least 17 peers at its Seattle data center [87]. Rather, cloud services are stuck with a "one size fits all" model because cloud providers select a single best route for all services, preventing cloud-based applications

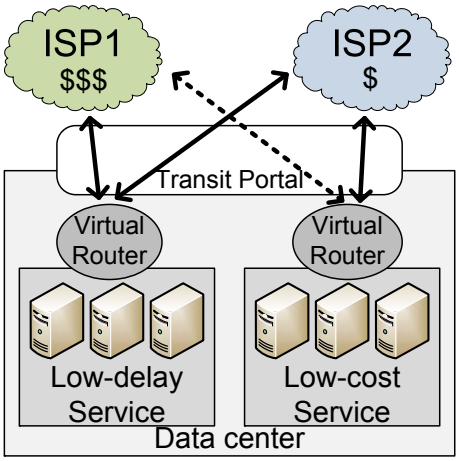


Figure 2.1: Connecting services though the Transit Portal.

from having any control over wide-area routing.

To give hosted services control over wide-area routing, we propose the *Transit Portal (TP)*, as shown in Figure 2.1. Each data center has a TP that gives each service the appearance of direct connectivity to the ISPs of its choice. Each service has a dedicated *virtual router* that acts as a gateway for the traffic flowing to and from its servers. The service configures its virtual router with its own policies for *selecting paths* to its clients and *announcing routes* that influence inbound traffic from its clients. By offering the abstraction of BGP sessions with each upstream ISP, the TP allows each service to capitalize on existing open-source software routers (including simple lightweight BGP daemons) without modifying its application software. That said, we believe extending TP to offer new, programmatic interfaces to distributed services is a promising avenue for future work.

Using the TP to control routing provides a hosted service significantly more control over the flow of its traffic than in today's data centers. In addition, the services enjoy these benefits without building their own network footprint, acquiring routeable address space, and negotiating with ISPs. These are hurdles that we ourselves faced in deploying TPs at several locations; the TP obviates the need for the services that it hosts to do the same. In addition, the TP simplifies operations for the ISPs by offloading the separate connections and relationships with each application and by applying packet and route filters to protect

them (and the rest of the Internet) from misconfigured or malicious services.

The design and implementation of the TP introduces several challenges. In the control plane, the TP must provide each virtual router the appearance of direct BGP sessions to its upstream ISPs. The TP must also forward outgoing packets to the right ISP and demultiplex incoming packets to the right virtual router. Our solutions to these problems must scale as the number of services increases. To solve these problems, we introduce a variety of techniques for providing layer-two connectivity, amortizing memory and message overhead, and filtering packets and routes. Our prototype implementation composes and extends open-source routing software—the Quagga software router for the control plane and the Linux kernel for the data plane—resulting in a system that is easy to deploy, maintain, and evolve. We also built a management system based on the GENI control framework [50] that automates the provisioning of new customers. The TP is deployed and operational at several locations.

This chapter makes the following contributions:

- We explain how flexible wide-area route control can extend the capabilities of existing hosting platforms.
- We present the design and implementation of Transit Portal and demonstrate that the system scales to many ISPs and clients.
- We quantify the benefits of TP by evaluating a DNS service that uses IP anycast and inbound traffic engineering on our existing TP deployment.
- We present the design and implementation of a management framework that allows hosted services to dynamically establish wide-area connectivity.
- We describe how to extend the TP to provide better forwarding performance and support a wider variety of applications (*e.g.*, virtual machine migration).

The remainder of the chapter is organized as follows. Section 2.1 explains how distributed services can make use of wide-area route control. Section 2.2 presents the design and implementation of the Transit Portal. Section 2.3 evaluates our three-site deployment

supporting an example service, and Section 2.4 evaluates the scalability and performance of our prototype. Section 2.5 presents our management framework, and Section 2.6 describes possible extensions to the TP. Section 2.8 compares TP to related work; we conclude in Section 2.9.

2.1 A Case for Wide-Area Route Control

We aim to give each hosted service the same level of routing control that existing networks have. Each service has its own virtual router that connects to the Internet through the *Transit Portal*, as shown in Figure 2.1. The Transit Portal allows each service to make a different decision about the best way to exchange traffic with its users. The Transit Portal also allows each service to announce prefixes selectively to different ISPs or send different announcements for the same IP prefix to control how inbound traffic reaches the announcing network.

2.1.1 How Route Control Helps Applications

This section describes three services that can benefit from having more control over wide-area routing and the ability to rapidly provision connectivity. Section 2.3 evaluates the first service we discuss—improving the reliability, latency, and load balacing traffic for distributed services—through a real deployment on Amazon's EC2. We do not evaluate the remaining applications, but we explain how they might be deployed in practice.

Reliable, low-latency distributed services. The Domain Name System (DNS) directs users to wide-area services by mapping a domain name to the appropriate IP address for that service. Service providers often use DNS for tasks like load balancing. Previous studies have shown that DNS lookup latency is a significant contributor to the overall latency for short sessions (*e.g.*, short HTTP requests). Thus, achieving reliability and low latency for DNS lookups is important. One approach to reducing DNS lookup latency is to move the authoritative DNS servers for a domain closer to clients using anycast. Anycast is a method

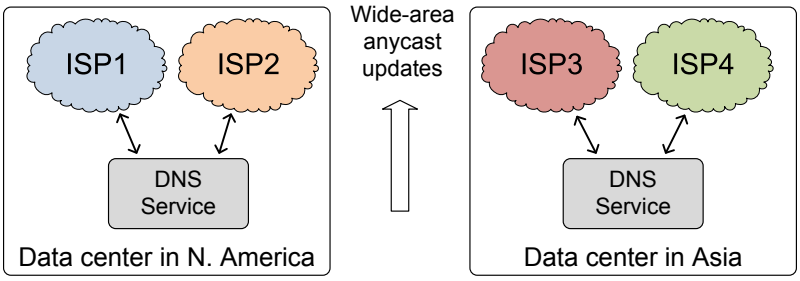


Figure 2.2: *Reliable, low-latency distributed services:* A service provider that hosts authoritative DNS for some domain may wish to provision both hosting and anycast connectivity in locations that are close to the clients for that domain.

where multiple distinct networks advertise the same IP prefix; client traffic then goes to one of these networks. Hosting authoritative name servers on an anycasted IP prefix can reduce the round-trip time to an authoritative name server for a domain, thus reducing overall name lookup time.

Although anycast is a common practice for DNS root servers, setting up anycast is a tall order for an individual domain: each domain that wants to host its own authoritative servers would need to establish colocation and BGP peering at multiple sites and make arrangements. Although a DNS hosting provider (*e.g.*, GoDaddy) could host the authoritative DNS for many domains and anycast prefixes for those servers, the domains would still not be able directly control their own DNS load-balancing and replication. Wide-area route control allows a domain to establish DNS-server replicas and peering in multiple locations and to change those locations and peering arrangements when load changes. Figure 2.2 shows such a deployment. We have deployed this service [99] and will evaluate it in Section 2.3.

Using routing to migrate services. Service providers such as Google commonly balance client requests across multiple locations and data centers to keep the latency for their services as low as possible. To do so, they commonly use the DNS to re-map a service name to a new IP address. Unfortunately, relying on DNS to migrate client requests requires the

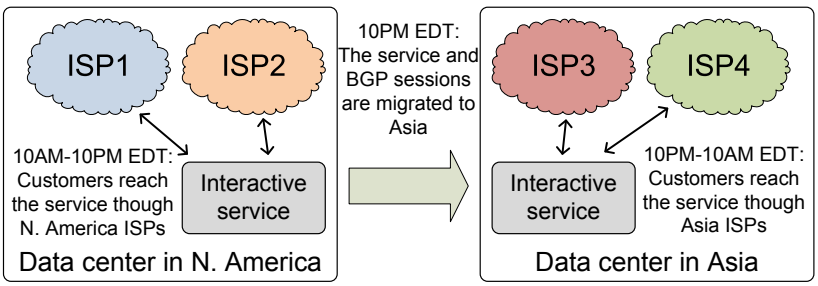


Figure 2.3: *Using routing to migrate services:* A service provider migrates a service from a data center in North America to one in Asia, to cope with fluctuations in demand. Today, service providers must use DNS for such migration, which can hurt user performance and does not permit the migration of a running service. A provider can use route control to migrate a service and re-route traffic on the fly, taking DNS out of the loop and enable migration of running services.

service provider to set low time-to-live (TTL) values on DNS replies. These low TTL values help a service provider quickly re-map a DNS name to a new IP address, but they also prevent the client from caching these records and can introduce significant additional latency; this latency is especially troublesome for short-lived sessions like Web search, where the DNS lookup comprises a large fraction of overall response time. Second, DNS-based re-mapping cannot migrate ongoing connections, which is important for certain services that maintain long-lived connections with clients (*e.g.*, VPN-based services). Direct wide-area route control allows the application provider to instead migrate services using routing: providers can migrate their services without changing server IP addresses by dynamically acquiring wide-area connections and announcing the associated IP prefix at the new data center while withdrawing it at the old one. Figure 2.3 shows how this type of migration can be implemented. This approach improves user-perceived performance by allowing the use of larger DNS TTL values and supporting live migration of long-lived connections.

Flexible peering & hosting for interactive applications. To minimize round-trip times, providers of interactive applications like gaming [5] and video conferencing [6, 7] aim to place servers close to their customers to users and, when possible, selecting the route corresponding to the lowest-latency path. When traffic patterns change due to flash-crowds,

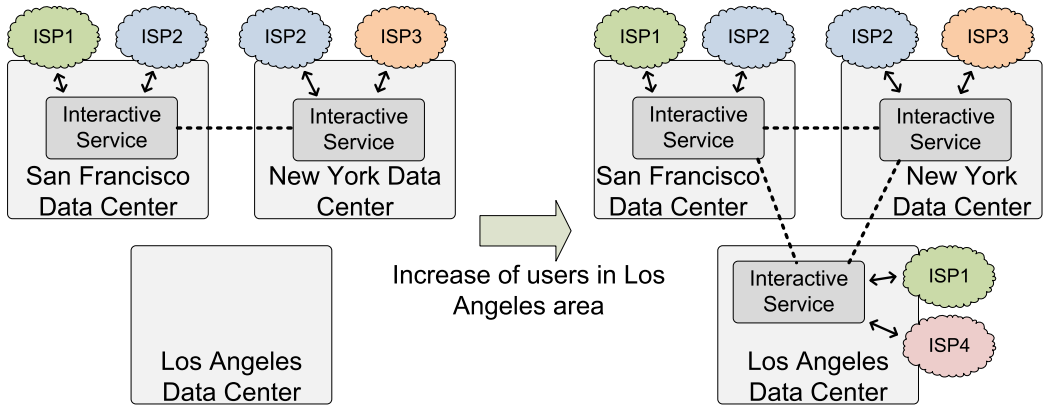


Figure 2.4: *Flexible peering and hosting for interactive applications:* Direct control over routing allows services to expand hosting *and upstream connectivity* in response to changing demands. In this example, a service experiences an increase in users in a single geographic area. In response, it adds hosting and peering at that location to allow customers at that location to easily reach the service.

diurnal fluctuations, or other effects, the application provider may need to rapidly reprovision both the locations of servers *and* the connectivity between those servers and its clients. Figure 2.4 shows an example of an interactive service that suddenly experiences a surge in users in a particular region. In this case, the hosting facility will not only need to provision additional servers for the interactive service provider, but it will also need to provision additional connectivity at that location to ensure that traffic to local clients enter and leave at that facility.

2.1.2 Deployment Scenarios

Cloud providers can provide direct control over routing and traffic to hosted applications. A cloud service such as Amazon's EC2 can use direct wide-area route control to allow each application provider to control inbound and outbound traffic according to its specific requirements. Suppose that two applications are hosted in the same data center. One application may be focused on maintaining low-cost connectivity, while the other may want to achieve low latency and good performance at any cost. Today's cloud services offer only "one size fits all" transit and do not provide routing control to each hosted service or application; the Transit Portal provides this additional control.

An OSP can re-provision resources and peering as demands change. Online service providers such as Google and Microsoft share a common infrastructure across multiple applications (*e.g.*, search, calendar, mail, video) and continually re-provision these resources as client demand shifts. Today, making application-specific routing decisions in a data center (as shown in Figure 2.1) is challenging, and re-routing clients to services in different data centers when demands change is even more difficult. The Transit Portal can provide each application in a data center control over routing and peering, allowing it to establish connectivity and select paths independently of the other properties. This function also makes service migration easier, as we describe in further detail below.

A researcher can perform experiments using wide-area routing. Although existing testbeds such as Emulab [42] allow researchers to operate their own wide-area networks, they generally do not offer flexible control over connectivity to the rest of the Internet. Different experiments will, of course, have different requirements for the nature of their connectivity and routing, and researchers may even want to experiment with the effects of different peering arrangements on experimental services.

2.2 Design and Implementation

This section describes the design and implementation of a Transit Portal (TP); Section 2.5 completes the picture by describing the management framework for a network of TPs. The TP extends and synthesizes existing software systems—specifically, the *Linux* kernel for the data plane and the *Quagga* routing protocol suite for the control plane. The rest of this section describes how our design helps the TP achieve three goals: (1) *transparent connectivity* between hosted services and upstream ISPs; (2) *scalability* to many hosted services and upstream ISPs; and (3) the ability to *protect* the rest of the Internet from accidental or malicious disruptions. Table 2.1 summarizes our design and implementation decisions and how they allow us to achieve the goals of transparent connectivity, scalability, and protection.

Table 2.1: TP design and implementation decisions.

Requirement	Decision	Implementation
Transparent Connectivity (Section 2.2.1)		
Different layer-two connections	Tunnels between TP and virtual router	Tunneling technologies supported by the Linux kernel
Transparent traffic forwarding	Isolated forwarding tables for ISPs	Virtual forwarding tables and policy routing in Linux
Scalability (Section 2.2.2)		
Scalable routing with the # of ISPs	Isolated routing tables for ISPs	BGP views feature in Quagga `bgpd` daemon
Scalable updates with # of clients	Shared route update computation	Peer-group feature in Quagga `bgpd` daemon
Scalable forwarding with # of ISPs	Policy/default routing	Modifications to the Quagga `bgpd` daemon
Protection (Section 2.2.3)		
Preventing IP address spoofing	Packet filtering on source IP address	Linux `iptables`
Preventing prefix hijacking	Route filtering on IP prefix	Quagga prefix filters
Limiting routing instability	Rate-limiting of BGP update messages	Route-flap damping in Quagga `bgpd` daemon
Controlling bandwidth usage	Traffic shaping on virtual interfaces	Linux `tc`

2.2.1 Transparent Connectivity

The TP gives client networks the appearance of direct data- and control-plane connectivity to one or more upstream ISPs. This transparency requires each client network to have a layer-two link and a BGP session for each upstream ISP that it connects to, even though the link and session for that client network actually terminate at the TP. The client's virtual routers are configured exactly as they would be if they connected directly to the upstream ISPs without traversing the Transit Portal. The TP has one layer-two connection and BGP session to each upstream ISP; this connection multiplexes both data packets and BGP messages for the client networks.

Different layer-two connections for different clients. Connecting to an upstream ISP normally requires the client to have a direct layer-two link to the ISP for carrying both BGP messages and data traffic. To support this abstraction, the TP forms a separate layer-two connection to the client for each upstream ISP. Our implementation uses the Linux 2.6 kernel support for IP-IP tunnels, GRE tunnels, EGRE tunnels, and VLANs, as well as UDP tunnels through a user-space OpenVPN daemon.

Transparent forwarding between clients and ISPs. The TP can use simple policy routing to direct traffic from each client tunnel to the corresponding ISP. Forwarding traffic from ISPs to clients, however, is more challenging. A conventional router with a single

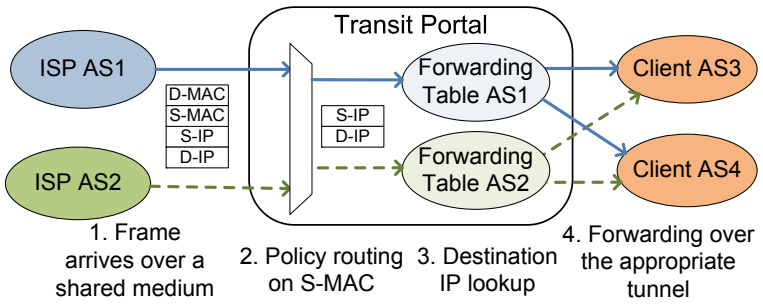


Figure 2.5: *Forwarding incoming traffic with TP:* When a packet arrives at the Transit Portal (Step 1), the TP uses the source MAC address (S-MAC) to demultiplex the packet to the appropriate IP forwarding table (Step 2). The lookup in that table (Step 3) determines the appropriate tunnel to the client network (Step 4).

```
1 # arp -a
2 router1.isp.com (1.1.1.1) at 0:0:0:0:0:11 on eth0
3 # iptables -A PREROUTING -t mangle -i eth0 -m mac\
4       --mac-source 0:0:0:0:0:11 -j MARK --set-mark 1
5 # ip rule add fwmark 1 table 1
```

Figure 2.6: Linux policy routing in TP de-multiplexes traffic into the appropriate forwarding table based on the packet's source MAC address. In this example, source address `0:0:0:0:0:11` de-multiplexes the packet into forwarding table `1`.

forwarding table would direct the traffic to the client prefix over a single link (or use several links in a round robin fashion if multipath routing is enabled.) As shown in Figure 2.5, though, the TP must ensure the packets are directed to the appropriate layer-two link—the one the client's virtual router associates with the upstream ISP. To allow this, the TP maintains a *virtual forwarding table* for each upstream ISP. Our implementation uses the Linux 2.6 kernel's support for up to 252 such tables, allowing the TP to support up to 252 upstream ISPs.

The TP can connect to an upstream ISP over a point-to-point link using a variety of physical media or tunneling technologies. We also intend to support deployment of Transit Portals at exchange points, where the TP may connect with multiple ISPs over a local area network via a single interface. Each ISP in such shared media setup sends layer-two frames using a distinct source MAC address; the TP can use this address to correctly identify the sending ISP. Figure 2.6 shows how such traffic de-multiplexing is configured

using policy routing rules. The ISP router has an IP address `1.1.1.1` with a MAC address `0:0:0:0:0:11` and a dedicated forwarding table, `1`. Line 1 shows the TP learning the MAC address of an upstream ISP when a new session is established. Then, lines 3–4 establish a policy-routing rule that redirects all the packets with this MAC address to a virtual forwarding table serving a new upstream ISP.

Transparency is another important goal for connectivity between client networks and the Transit Portal. In other words, a client network's connection to the TP should appear as though it were directly connected to the respective upstream networks. In the control plane, achieving this goal involves (1) removing the appearance of an extra AS hop along the AS path; and (2) passing BGP updates between client networks and upstreams as quickly as possible. The first task is achieved with the `remove-private-as rewrite` configuration (line 10 in Figure 2.7), and the second task is achieved by setting the advertisement interval to a low value (line 18 in Figure 2.7).

The Transit Portal supports clients regardless of whether they have a public or private AS number. To ensure transparency for the clients with a public AS number, the TP forwards the updates from such clients unmodified. Updates from clients with private AS numbers require rewriting.

2.2.2 Scalability

The TP maintains many BGP sessions, stores and disseminates many BGP routes, and forwards packets between many pairs of clients and ISPs. Scaling to a large number of ISPs and clients is challenging because each upstream ISP announces routes for many prefixes (*i.e.*, 300,000 routes); each client may receive routes from many (and possibly all) of these ISPs; and each client selects and uses routes independently. We describe three design decisions that we used to scale routing and forwarding at the TP: BGP views, peer groups, and default routing.

Scalable routing tables using BGP views. Rather than selecting a single best route for each destination prefix, the TP allows each service to select from the routes learned from all the upstream ISPs. This function requires the Transit Portal to disseminate routes from each ISP to the downstream clients, rather than selecting a single best route and could be achieved by having the TP run a separate instance of BGP for each upstream ISP, with BGP sessions with the ISP and each of the clients. Unfortunately, running multiple instances of BGP, each with its own process and associated state, would be expensive. Instead, the TP runs a single BGP instance with multiple "BGP views", each with its own routing table and decision process, for each upstream ISP. Using BGP views prevents the TP from comparing routes learned from different ISPs, while still capitalizing on opportunities to store redundant route information efficiently. Any downstream client that wants to receive routing messages from a specific upstream ISP need only establish a BGP session to the associated view. Our implementation uses the BGP `view` feature in Quagga; in particular, Figure 2.7 shows the configuration of a single "view" (starting in line 3) for upstream ISP 1. Section 2.4.2 shows that using BGP views in Quagga allows us to support approximately 30% more upstream ISPs with the same memory resources compared to the number of supported ISPs using conventional BGP processes.

Scalable updates to clients with BGP peer groups. Upon receiving a BGP update message from an upstream ISP, the TP must send an update message to each client that "connects" to that ISP. Rather than creating, storing, and sending that message separately for each client, the TP maintains a single BGP table and constructs a common message to send to all clients. Our implementation achieves this goal by using the `peer-group` feature in Quagga, as shown in Figure 2.7; in particular, line 14 associates `Client A` (CA) with the `peer-group` View1 for upstream ISP 1, as defined in lines 17–20. Note that although this example shows only one peer-group member, the real benefit of peer groups is achieved when multiple clients belong to the same group. Section 2.4.3 shows that peer-groups

```
bgp multiple-instance
!
router bgp 47065 view Upstream1
  bgp router-id 47.0.0.65
  bgp forwarding-table 1
  bgp dampening

  ! Connection to Upstream ISP
  neighbor 1.1.1.1 remote-as 1
  neighbor 1.1.1.1 remove-private-AS rewrite
  neighbor 1.1.1.1 attribute-unchanged as-path med

  ! Connection to Downstream Client
  neighbor 2.2.2.2 peer-group View1
  neighbor 2.2.2.2 remote-as 2
  neighbor 2.2.2.2 route-map CA-IN in
  neighbor View1 peer-group
  neighbor View1 advertisement-interval 2
  neighbor View1 attribute-unchanged as-path med
  neighbor View1 local-as 1 no-prepend
!
ip prefix-list CA seq 5 permit 2.2.2.0/24
!
route-map CA-IN permit 10
  match ip address prefix-list CA
!
```

Figure 2.7: Quagga configuration.

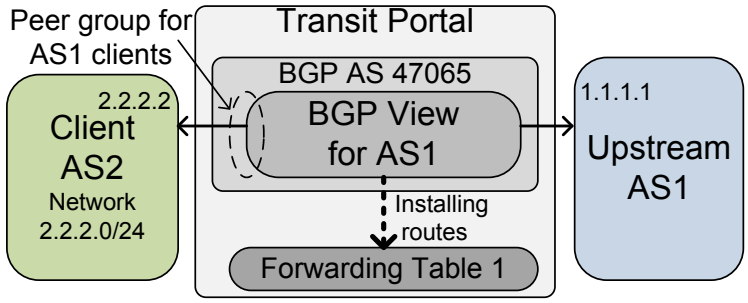


Figure 2.8: Control-plane setup.

reduce CPU consumption threefold.

Smaller forwarding tables with default routes and policy routing. Despite storing and exchanging many BGP routes in the control plane, the Transit Portal should try to limit the amount of data-plane state for fast packet forwarding. To minimize data-plane state, the TP does *not* install all of the BGP routes from each BGP view in the kernel forwarding tables. Instead, the TP installs the smallest amount of state necessary for custom packet forwarding to and from each client. On each virtual forwarding table, the TP stores only

a default route to an upstream ISP associated with that table (to direct clients' outbound traffic through the ISP) and the BGP routes announced by the clients themselves (to direct inbound traffic from the ISP to the appropriate client). As shown in Section 2.4.2, this arrangement allows us to save about one gigabyte of memory for every 20 upstream ISPs. To selectively install only the routes learned from clients, rather than all routes in the BGP view, we make modifications to Quagga.

2.2.3 Protection

The TP must protect other networks on the Internet from misbehavior such as IP address spoofing, route hijacking or instability, or disproportionate bandwidth usage.

Preventing spoofing and hijacking with filters. The TP should prevent clients from sending IP packets or BGP route announcements for IP addresses they do not own. The TP performs ingress packet filtering on the source IP address and route filtering on the IP prefix, based on the client's address block(s). Our implementation filters packets using the standard `iptables` tool in Linux and filters routes using the `prefix-list` feature, as shown in lines 16 and 22-26 of Figure 2.7. In addition to filtering prefixes the clients do not own, the TP also prevents clients from announcing smaller subnets (*i.e.*, smaller than a /24) of their address blocks. Smaller subnets are also filtered by default by most of the Internet carriers. Section 2.6 describes how TP can overcome this limitation.

Limiting routing instability with route-flap damping. The TP should also protect the upstream ISPs and the Internet as a whole from unstable or poorly managed clients. These clients may frequently reset their BGP sessions with the TP, or repeatedly announce and withdraw their IP prefixes. The TP uses route-flap damping to prevent such instability from affecting other networks. Our implementation enables route-flap damping (as shown in line 6 of Figure 2.7) with the following parameters: a half-life of 15 minutes, a 500-point penalty, a 750-point reuse threshold, and a maximum damping time of 60 minutes. These

settings allow client to send the original update, followed by an extra withdrawal and an update, which will incur penalty of 500 points. Additional withdrawals or updates in a short timeframe will increase the penalty above reuse threshold and the route will be suppressed until the penalty shrinks to 750 points (the penalty halves every 15 minutes). There is no danger that one client's flapping will affect other clients, as route damping on the Internet operates separately for each announced route.

Controlling bandwidth usage with rate-limiting. The TP should prevent clients from consuming excessive bandwidth, to ensure that all clients have sufficient resources to exchange traffic with each upstream ISP. The TP prevents bandwidth hoarding by imposing rate limits on the traffic on each client connection. In our implementation, we use the standard `tc` (traffic control) features in Linux to impose a maximum bit rate on each connection to clients.

2.3 Deployment

We have deployed Transit Portals in five locations. The TPs are deployed in Atlanta GA, Clemson NC, Madison WI, Princeton NH, and Seattle WA. All Transit Portals are deployed in universities and research labs, whose networks act as a sole ISP in each location. Each ISP also provides full transit for our prefix and AS number. We are actively expanding this deployment: We are engaging with operators at two European research institutions and with one commercial operator in the U.S. to deploy more Transit Portals, and we are planning to expand our Seattle installation to connect to more ISPs.

The TPs advertise BGP routes using origin AS 47065 and IP prefix 168.62.16.0/21. Clients currently use a private AS number that the TP translates to the public AS number, 47065, before forwarding an update. Clients can also obtain their own AS number, in which case the TP re-advertises the updates without modification.

This section presents the deployment of a distributed, anycasted DNS service, as we

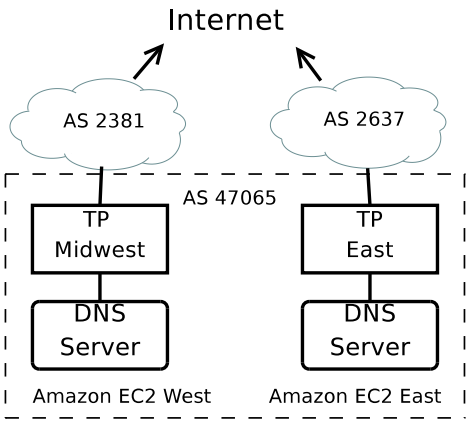


Figure 2.9: The TP IP anycast experiment setup.

described in Section 2.1, that uses the TP for traffic control, similar to the service we described in Section 2.1 (Figure 2.2). In our evaluation of this deployment, we demonstrate two distinct functions: (1) the ability to load balance inbound and outbound traffic to and from the DNS service (including the ability to control the number of clients that communicate with each replica); and (2) the ability to reduce latency for specific subsets of clients with direct route control.

2.3.1 DNS With IP Anycast

In this section, we show how the TP delivers control and performance improvements for applications that can support IP anycast, such as anycast DNS resolution. The TP allows an IP anycast service to: (1) react to failures more quickly than using DNS re-mapping mechanisms, (2) load-balance inbound traffic, and (3) reduce the service response time. We explain the experiment setup and the measurements that show that adding IP anycast to services running on Amazon EC2 servers can improve latency, failover, and load-balance.

We deploy two DNS servers in Amazon EC2 data centers: one in the US-East region (Virginia) and another in the US-West region (Northern California). The servers are connected to two different TP sites and announce the same IP prefix to enable IP anycast routing as shown in Figure 2.9. The US-East region is connected to AS 2637 as an upstream provider, while the US-West region is connected to AS 2381 as its upstream provider. We

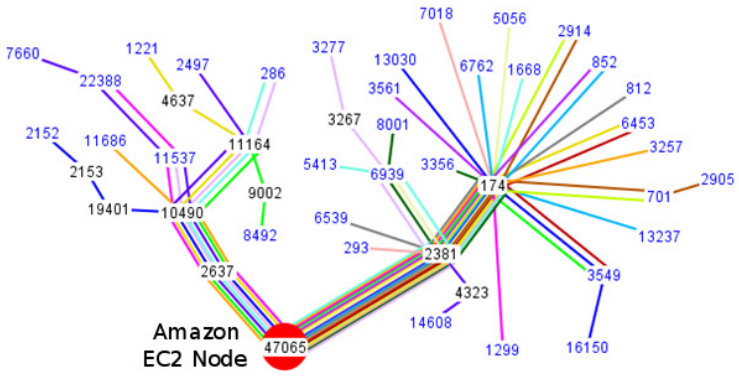


Figure 2.10: AS-level paths to an EC2 service sitting behind the Transit Portal (AS 47065), as seen in RouteViews.

measure the reachability and delay to these DNS servers by observing the response time to the IP anycast address from approximately 600 clients on different PlanetLab [91] nodes. Because our goal is to evaluate the scenario where the TP is collocated with a cloud computing facility, we adjust the measurements to discount the round-trip delay between the TP and the Amazon EC2 data centers.

The main provider of both upstreams is Cogent (AS 174), which by default prefers a downstream link to AS 2381. Cogent publishes a set of rules that allows Cogent's clients (*e.g.*, AS 2381, AS 2637, and their respective clients) to affect Cogent's routing choices [39]. The DNS service hosted on an Amazon host runs a virtual router and thus can apply these rules and control how incoming traffic ultimately reaches the service.

Figure 2.10 shows a capture from the BGPlay tool [1], which shows the initial routing state with the original BGP configuration. Most of the Internet paths to AS 47065 traverse Cogent (AS 174), which in turn prefers AS 2381 to forward the traffic to the client. Note that the client is configured with private AS number, but the TPs rewrite the updates before re-advertising them to the Internet. This rewriting causes the routers on the Internet to observe prefixes from as if they were announced by AS 47065.

Failover. Today's Internet applications use DNS name re-mapping to shift services to active data centers in the case of a data center or network failure. DNS name re-mapping is

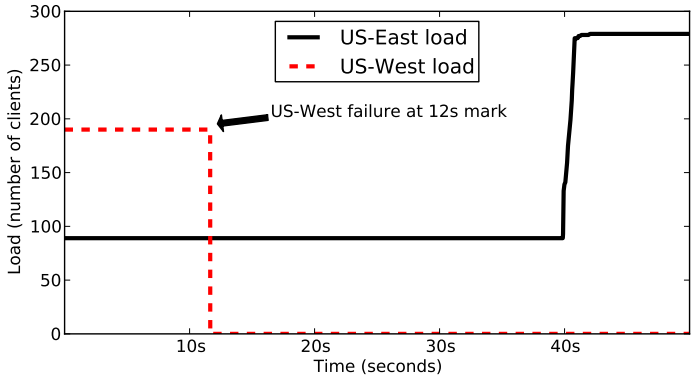


Figure 2.11: Failover behavior with two IP anycast servers using the TP.

a relatively slow process because it requires the DNS entries to expire in DNS caches across the Internet. Applications that support IP anycast can rely on the routing infrastructure to route traffic to active data centers. In our experiment, we fail the server deployed in the US-West region and observe how quickly the clients converge to the US-East region.

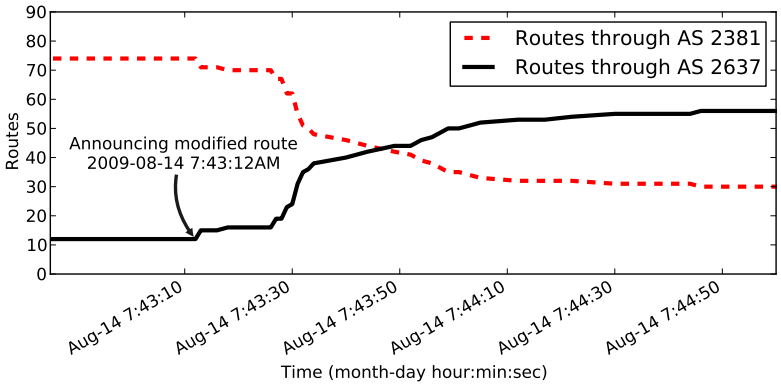
Figure 2.11 shows how the load changes as we introduce failure. After twelve seconds, we fail the US-West deployment and stop receiving requests at that site. After approximately 30 seconds, the routing infrastructure reroutes the traffic to the US-East site. The reaction to failure was automatic, requiring no monitoring or intervention from either the application or the TP operators.

Inbound traffic control. Assume that the DNS service would prefer most of its traffic to be served via AS 2637, rather than AS 2381. (The client network might prefer an alternate route as a result of cost, security, reliability, delay, or any other metric.) The Transit Portal clients can apply BGP communities to affect how upstream ISPs routes to its customers. On August 14, we changed the client configuration as shown in Figure 12(a) to add the BGP community `174:10` to a route, which indicates to one of the upstream providers, Cogent (AS 174), to prefer this route less than other routes to the client network.

To see how quickly the Internet converges to a new route, we analyze the route information provided by RouteViews. Figure 12(b) shows the convergence of the Internet routes

```
1 router bgp 65000
2   neighbor 10.1.0.1 route-map OUT-2381 out
3 !
4 route-map OUT-2381 permit 10
5   match ip address prefix-list OUR
6   set community 174:10
```

(a) Route map.



(b) Route convergence after applying the route map.

Figure 2.12: Load balance: Applying a route map to outbound advertisements to affect incoming traffic.

to a new upstream. The dashed line shows the number of networks on the Internet that use the AS 2381 link, while the plain line shows the number of networks that use the AS 2637 link to reach the client hosted in the Amazon data center. (Note that the number of routes corresponds only to the routes that we collected from RouteViews.)

IP anycast application performance. We evaluate three DNS service scenarios: (1) US-East only, (2) US-West only, and (3) both servers using IP anycast routing. We measure the delay the PlanetLab clients observe to the IP anycast address in each of these scenarios. Using IP anycast should route each client to the closest active data center.

Table 2.2 shows the results of these experiments. Serving DNS using IP anycast improves response time by 4-8 milliseconds compared to serving from either of the sites separately. The improvement is not significant in our setup, since the Midwest and East Coast TP deployments are not far from each other. We expect more improvement when IP

Table 2.2: DNS anycast deployment with the TP. Average round trip time to the service and fraction of the load to each of the sites.

	Avg. Delay	US-East	US-West
US-East	102.09ms	100%	0%
US-West	98.62ms	0%	100%
Anycast	94.68ms	42%	58%

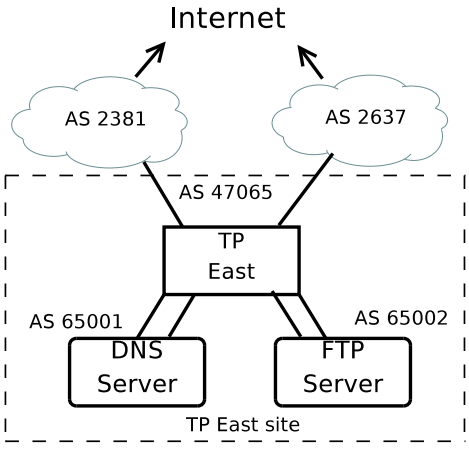


Figure 2.13: Outbound TE experiment setup with the TP.

anycast is used from more diverse locations.

2.3.2 Outbound Traffic Control

We now show how two different services in a single data center can apply different outbound routing policies to choose different exit paths from the data center. Figure 2.13 shows the demonstration setup. DNS and FTP services run virtual routers configured as AS 65001 and AS 65002 respectively; both services and the Transit Portal are hosted at the same site as the ISP with an AS 2637. The ISP with an AS 2381 is in a different location and, for the sake of this experiment, the TP routes connections to it via a tunnel.

Without a special configuration, the services would choose the upstream ISP based on the shortest AS path. In our setup, we use the `route-map` command combined with a `set local-preference` setting to configure AS 65001 (DNS service) to prefer AS 2637 as an upstream and AS 65002 (FTP service) to prefer AS 2381 as an upstream. Figure 2.14 shows how the traceroutes to a remote host differ because of this configuration. The first

```
1 AS65001-node:~# traceroute -n -q 1 -A 133.69.37.5
2 traceroute to 133.69.37.5
3  1  10.0.0.1 [*]  0 ms
4  2  143.215.254.25 [AS2637]  0 ms
5 [skipped]
6  8  203.181.248.110 [AS7660]  187 ms
7  9  133.69.37.5 [AS7660]  182 ms
```

(a) Traceroute from AS 65001 client.

```
1 AS65002-node:~# traceroute -n -q 1 -A 133.69.37.5
2 traceroute to 133.69.37.5
3  1  10.1.0.1 [*]  23 ms
4  2  216.56.60.169 [AS2381] 23 ms
5 [skipped]
6  9  192.203.116.146 [*]  200 ms
7  10  133.69.37.5 [AS7660]  205 ms
```

(b) Traceroute from AS 65002 client.

Figure 2.14: Traceroute from services co-located with TP East and AS 2637.

hop in AS 65001 is a local service provider and is less than one millisecond away. The AS 65002 tunnels are transparently switched through a local TP and terminated at the remote AS 2381, which introduces additional delay.

2.3.3 Performance Optimization

The TP can be used to optimize the Internet service access performance. We simulate a video content provider with video streaming services running in cloud sites at the Princeton

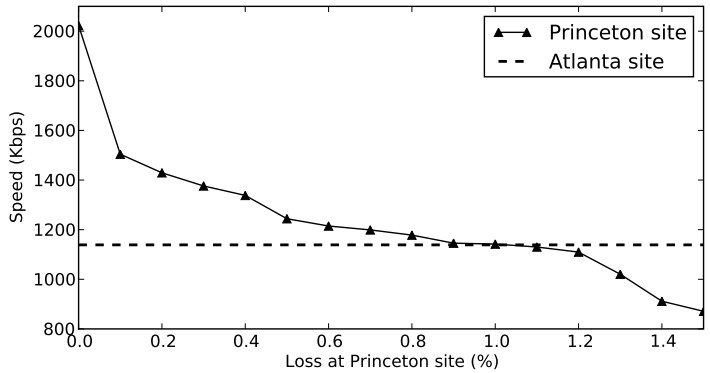


Figure 2.15: Average access speed from U.S. North East as packet loss is introduced in TP at Princeton site.

and the Atlanta locations. Bandwidth measurements show that the Princeton site offers better speed for clients in the northeast U.S., while the Atlanta site is preferred for the southeast U.S.

Assume that, due to periodic congestion, Princeton experiences intermittent packet loss every day around noon. Because the packet loss is intermittent, the application operator may be reluctant to use DNS to re-map the clients. Instead of DNS, the operator can use TP for rerouting when packet loss is detected. Figure 2.15 shows the average service access speed from the clients in the northeast as the loss at the Princeton site is increasing. As Princeton reaches a 1.1% packet loss rate, the Atlanta site, with its baseline speed of 1140 Kbps, becomes a better choice. Application operators then can use the methods described in Section 2.3.1 and 2.3.2 to reroute their applications when they observe losses in Princeton higher than 1.1%.

2.4 Scalability Evaluation

This section performs micro-benchmarks to evaluate how the Transit Portal scales with the number of upstream ISPs and client networks. Our goal is to demonstrate the feasibility of our design by showing that a TP that is implemented in commodity hardware can support a large number of upstream ISPs and downstream clients. Our evaluation quantifies the number of upstream and client sessions that the TP can support and shows how various design decisions from Section 2.2 help improve scalability. We first describe our evaluation setup; we then explore how the number of upstream ISPs and downstream client networks affect the TP's performance for realistic routing workload. Our findings are unsurprising but comforting: A single TP node can support tens of upstream ISPs and hundreds of client networks using today's commodity hardware, and we observe it scaling linearly to the load.

2.4.1 Setup

Data. To perform repeatable experiments, we constructed a BGP update dataset, which we used for all of our scenarios. We used BGP route information provided by RIPE Route

Table 2.3: RIPE BGP data set for September 1, 2009.

AS	Prefixes	Updates	Withdrawals
RCCN (1930)	291,996	267,207	15,917
Tinet (3257)	289,077	205,541	22,809
RIPE NCC (3333)	293,059	16,036,246	7,067
Global Crossing (3549)	288,096	883,290	68,185
APNIC NCC (4608)	298,508	589,234	9,147
APNIC NCC (4777)	294,387	127,240	12,233
NIX.CZ (6881)	290,480	150,304	11,247
AT&T (7018)	288,640	1,116,576	904,051
Hutchison (9304)	296,070	300,606	21,551
IP Group (16186)	288,384	273,410	47,776

Information Service (RIS) [96]. RIPE RIS provides two types of BGP update data: 1) BGP table dumps, and 2) BGP update traces. Each BGP *table dump* represents a full BGP route table snapshot. A BGP *update trace* represents a time-stamped arrival of BGP updates from BGP neighbors. We combine the dumps with the updates: Each combined trace starts with the stream of the updates that fill in the BGP routing table to reflect a BGP dump. The trace continues with the time-stamped updates as recorded by the BGP update trace. When we replayed this trace, we honored the inter-arrival intervals of the update trace.

Table 2.3 shows our dataset, which has BGP updates from 10 ISPs. The initial BGP table dump is taken on September 1, 2009. The updates are recorded in 24-hour period starting on midnight September 2 and ending at midnight September 3 (UTC). The average BGP table size is 291,869.1 prefixes. The average number of updates during a 24-hour period is 1,894,474.3, and the average number of withdrawals is 111,998.3. There are more announcements than withdrawals (a withdrawal occurs only if there is no alternate route to the prefix).

The data set contains two upstream ISPs with an unusually high number of updates: AS 3333 with more than 16 million updates, and AS 7018 with more than 900,000 withdrawals. It is likely that AS 3333 or its clients use reactive BGP load-balancing. In AS 7018, the likely explanation for a large number of withdrawals is a misconfiguration, or a flapping link. In any case, these outliers can stress the Transit Portal against extreme

load conditions.

Test environment. We replayed the BGP updates using the `bgp_simple` BGP player [25]. The `bgp_simple` player is implemented using Perl `Net::BGP` libraries. We modified the player to honor the time intervals between the updates.

Unless otherwise noted, the test setup consists of five nodes: Two nodes for emulating clients, two nodes for emulating upstream ISPs, and one node under test, running the Transit Portal. The test node has two 1 Gbps Ethernet interfaces, which are connected to two LANs: one LAN hosts client-emulating nodes, the other LAN hosts upstream-emulating nodes. Each node runs on a Dell PowerEdge 2850, with a 3 Ghz dual-core 64-bit Xeon CPU and 2 GB of RAM. The machines run Fedora 8 Linux.

When we measured CPU usage for a specific process, we used the statistics provided by `/proc`. Each process has a `jiffies` counter, which records the number of system ticks the process used so far. For each test, we collect jiffies at five-second intervals over three minutes and the compute average CPU usage in percent. The `vmstat` utility provides the overall memory and CPU usage.

2.4.2 Memory Usage

Upstream sessions. Using a commodity platform with 2 GB of memory, TP scales to a few dozen of upstream ISPs. Figure 2.16 shows how the memory increases as we add more upstream ISPs. When TP utilizes separate BGP processes, each upstream ISP utilizes approximately 90MB of memory; using BGP views each upstream ISP utilizes approximately 60MB of memory. Data plane memory usage, as shown in Figure 2.17, is insignificant when using our forwarding optimization.

Downstream sessions. Each session to a downstream application consumes approximately 10MB of memory. For example, given 20 upstream ISPs, a client "subscribing" to all of them will consume 200MB. Upgrading the TP machine to 16GB of memory would

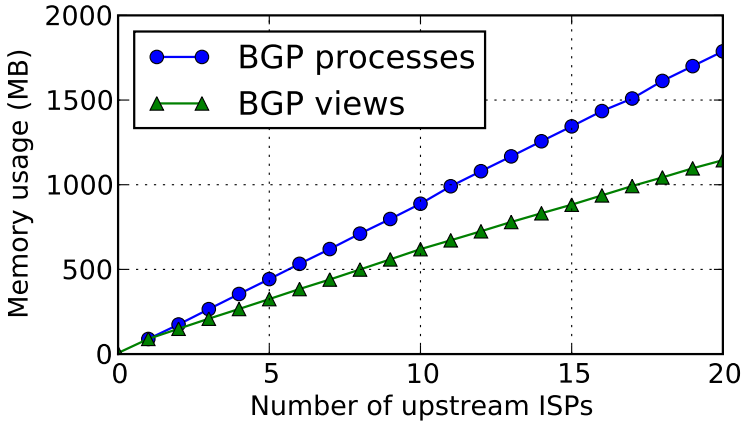


Figure 2.16: The TP control plane memory use.

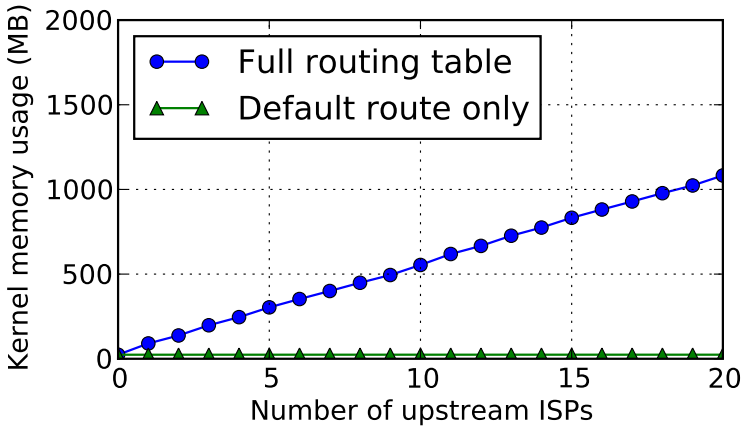


Figure 2.17: The TP data plane memory use.

easily support 20 upstream ISPs with more than a hundred clients subscribing to an average of 10 ISPs. The clients use only a small amount of memory in the data plane. The TP ensures forwarding only to the prefixes clients own or lease.

2.4.3 CPU Usage and Propagation Time

The main users of TP CPU are a BGP scan process, which scans the routes for the changes in reachability information, and BGP update parsing process, which parses the updates which arrive intermittently at a rate of approximately 2 million updates per day.

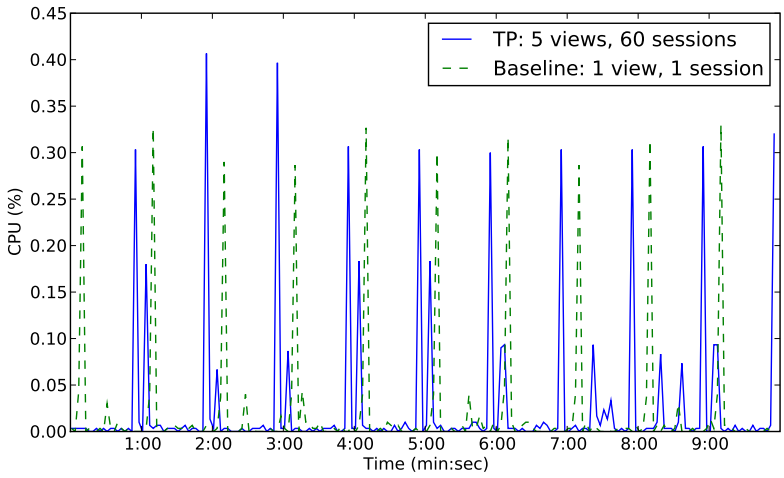


Figure 2.18: The TP CPU usage over time (average taken every 3 seconds).

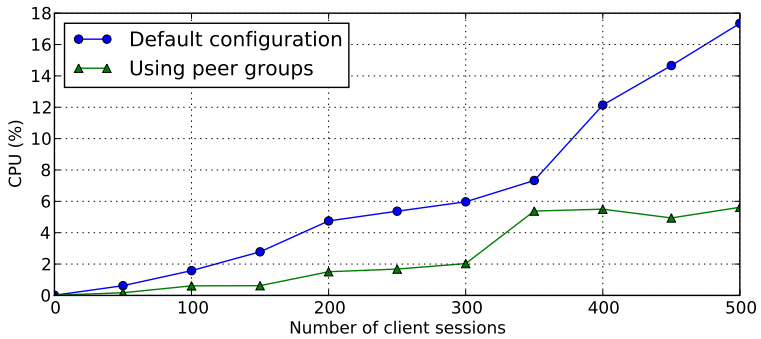


Figure 2.19: The TP CPU usage while adding client sessions.

Figure 2.18 shows the timeseries of the CPU usage of two BGP processes as they perform routine tasks. Routing updates for both processes arrive from five different ISPs according to their traces. The baseline uses a default Quagga configuration with one routing table, and one client. The Transit Portal configuration terminates each ISP at a different virtual routing table and connects 12 clients (which amounts to a total of 60 client sessions). The TP configuration, on average consumes 20% more CPU than a baseline configuration; most of this load overhead comes from maintaining more client sessions. The spikes in both plots correspond to a BGP scan that occurs every 60 seconds.

The TP can easily support hundreds of client BGP sessions. Figure 2.19 shows CPU

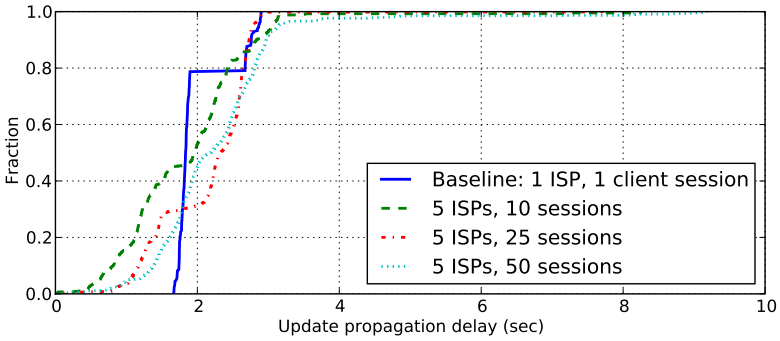


Figure 2.20: Update propagation delay through the TP.

usage as more client sessions are added. Two plots shows CPU usage using the default client configuration and CPU usage using a client configuration with a peer-group feature enabled. While conducting this experiment, we add 50 client sessions at a time and measure CPU load. We observe fluctuations in CPU use since at each measurement the number of updates passing the TP is slightly different. Nevertheless the trend is visible and each one hundred of client sessions increase CPU utilization by approximately a half of a percent.

Figure 2.20 shows the update propagation delays though the TP. The baseline configuration uses minimal configuration of Quagga with advertisement interval set to 2 seconds. Other configurations reflect the setup of five upstream providers with 10, 25, and 50 sessions. Approximately 40% of updates in the setup with 10 sessions are delivered within 1.6 seconds, while the baseline configuration seems to start deliver updates only at around 1.7 seconds due to the grouping of updates at the TP. A single upstream ISP sends updates in batches and each batch is subject to the configured two-second advertisement interval. When multiple upstream ISPs are configured, more updates arrive at the middle of advertisement interval and can be delivered as soon as it expires.

2.5 Framework for Provisioning Resources

The TP service provides an interface to the clients of existing hosting facilities to provision wide-area connectivity. In this section, we describe the design and implementation of

```
<rspec type="advertisement" >

 <node virtual_id="tp1">
   <node_type type_name="tp">
     <field key="encapsulation" value="gre"/>
     <field key="encapsulation" value="egre"/>
     <field key="upstream_as" value="1"/>
     <field key="upstream_as" value="2"/>
     <field key="prefix" count="3" length="24"/>
   </node_type>
 </node>
```

Figure 2.21: *Resource advertisement:* In Step 0 of resource allocation (Figure 2.22), the TP's component manager advertises available resources. This example advertisement says that the TP supports both GRE and EGRE encapsulation, has connections to two upstream ASes, and has three /24 prefixes available to allocate.

this management plane. We first discuss the available resources and how they are specified. Next, we describe the process for clients to discover and request resources. Then, we discuss how we have implemented the management plane in the context of the GENI control framework [50]. In the future, the management plane will also control the hosting resources, and provide clients a single interface for resource provisioning.

2.5.1 Resources and Their Specification

The current resource allocation framework tracks *numbered* and *network* resources. The numbered resources include the available IP address space, the IP prefixes assigned to each client network, the AS number (or numbers) that each client is using to connect to the TPs, and which IP prefix will be advertised from each location. The framework must also keep track of whether a client network has its own IP prefix or AS number. Network resources include the underlying physical bandwidth for connecting to clients, and bandwidth available to and from each upstream ISP. Management of hosting resources, at this stage, is left for the client networks to handle.

The available resources should be specified in a consistent, machine-readable format.

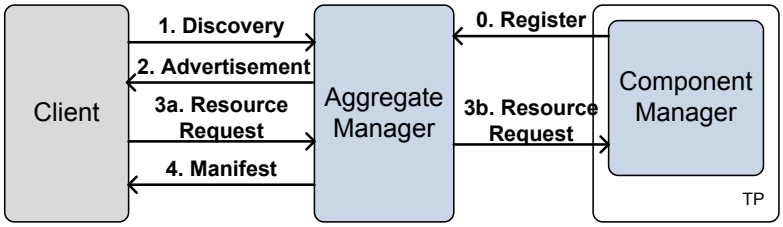


Figure 2.22: TP resource allocation process.

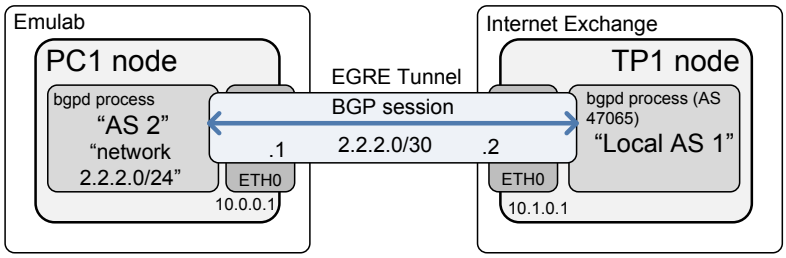


Figure 2.23: TP topology resulting from the resource request.

Our implementation represents resources using XML. Figure 2.21 shows an example advertisement, which denotes the resources available at one TP that offers two upstream connections, as indicated by lines 7–8 in the advertisement. The advertisement also indicates that this TP has three prefixes available for clients (line 9) and can support both GRE and EGRE tunneling (lines 5–6).

2.5.2 Discovering and Requesting Resources

Each Transit Portal runs a *component manager* (CM) that tracks available resources on the node. To track available capacity between TPs, or on links between virtual hosting facilities, the service uses an *aggregate manager* (AM). The aggregate manager maintains inventory over global resources by aggregating the available resources reported by the component managers. It also brokers client requests by contacting individual CMs, as shown in Figure 2.22.

Clients can discover and request resources using a supplied command line tool `en-client.py`. The tool can issue resource discovery and resource reservation requests to a hard-coded AM address as shown in Section 2.3.2.

```
<rspec type="request" >
 <node virtual_id="tp1">
   <node_type type_name="tp">
     <field key="upstream_as" value="1" />
     <field key="prefix" count="1">
   </node_type>
 </node>
 <link virtual_id="link0">
   <link_type name="egre">
     <field key="ttl" value="255">
   </link_type>
   <interface_ref virtual_node_id="tp1" />
   <interface_ref virtual_node_id="pc1"
     tunnel_endpoint="10.0.0.1" />
 </link>
</rspec>
```

(a) The *resource request* specifies the client's tunnel endpoint, `10.0.0.1`, and asks for an EGRE tunnel, as well as an IP prefix and upstream connectivity to AS 1.

```
<rspec type="manifest" >
  <node virtual_id="tp1">
    <node_type type_name="tp">
      <field key="upstream_as" value="1" />
      <field key="prefix" count="1" value="2.2.2.0/24" />
    </node_type>
  </node>
  <link virtual_id="link0">
    <link_type name="egre">
      <field key="ttl" value="255" />
    </link_type>
    <interface_ref virtual_node_id="tp1"\
      tunnel_endpoint="10.1.0.1"\
      tunnel_ip="2.2.2.2/30" />
    <interface_ref virtual_node_id="pc1"\
      tunnel_endpoint="10.0.0.1"\
      tunnel_ip="2.2.2.1/30" />
  </link>
</rspec>
```

(b) The *manifest* assigns an IP prefix to the network, `2.2.2.0/24`, and specifies the parameters for the tunnel between PC1 and TP1.

Figure 2.24: The *resource request* (Step 3) and *manifest* (Step 5) of the resource allocation process, for an example topology.

Before clients can request resources, the AM must know about resources in all TP installations. Each component manager *registers* with the AM and provides the list of the available resources, such as the list of upstream ISPs and available IP prefixes (Step 0 in Figure 2.22). To request resources, a client first issues a *discovery* call to an AM (Step 1). The AM replies with *advertisement*, which describes resources available for reservation (Step 1), such as the example in Figure 2.21. After receiving the resource advertisement, a client can issue a *resource request* (Step 3), such as the example in Figure 24(a). If the resources are available, the AM issues the *reservation request* to TP1 (Step 4) and responds with a *manifest* (Step 5), which is annotated version of the *request* providing the missing information necessary to establish the requested topology, as shown in Figure 24(b). The AM also provides sample client configuration excerpts with the manifest to streamline client configuration setup. The client uses the manifest to configure its end of the links and sessions, such as the configuration of PC1 in Figure 2.23.

2.5.3 Implementation

We implement the provisioning framework in the spirit of the Slice-based Facility Architecture (SFA) [8]. This management plane approach is actively developed by projects in the GENI [50] initiative, such as ProtoGENI [93]. SFA is a natural choice because our intermediate goal is to integrate the TP with testbeds like ProtoGENI [93] and VINI [111]. We use the *Twisted* event-driven engine libraries written in *Python* to implement the management plane components.

The primary components of the SFA are the Component Manager (CM) and Aggregate Manager (AM) as introduced before. The interface between the AM and CM is implemented using XML-RPC calls. The client interacts with the AM though a front-end, such as the Emulab or PlanetLab Web site, which in turn contacts the AM using XML-RPC or through the supplied client script.

Currently, access control to the AM is controlled with static access rules that authenticate clients and authorize or prevent a client from instantiating resources. To support more dynamic access control, we plan to expand the AM and CM to support security credentials, which will enable us to inter-operate with existing facilities (*e.g.*, PlanetLab, VINI, GENI) without using static access rules. We also plan to extend the resource management to include slice-based resource allocation and accounting.

2.6 Extensions to the Transit Portal

The TP is extensible. In the future, we plan to add support for lightweight clients who don't want to run BGP, support for smaller IP prefixes, support for backhaul between different TP sites, extensions for better scalability using hardware platforms for the data plane, and extensions for better routing stability in the face of transient client networks.

Support for lightweight clients. Some client networks primarily need to control traffic but do not necessarily need to run BGP between their own networks and the transit portal. In these cases, a client could use the existence or absence of a tunnel to the TP to signal to the TP whether it wanted traffic to enter over a particular ingress point. When the client network brings up a tunnel, the TP could announce the prefix over the appropriate ISP. When the client brings the tunnel down, the TP withdraws the prefix. As long as the tunnels are up, the client is free to choose an outgoing tunnel to route its traffic.

Support for small IP prefixes. Many client networks may not need IP addresses for more than a few hosts; unfortunately, these client networks would not be able to advertise their own IP prefix on the network, as ISPs typically filter IP prefixes that are longer than a /24 (*i.e.*, subnetworks with less than 256 addresses). The TP could allow client networks with fewer hosts to have BGP-like route control without having to advertise a complete /24 network. Clients for such networks would have full control of outgoing route selection and limited control for influencing incoming routes.

Better scalability. The scalability of the TP data plane can be further improved in two ways: (1) running multiple TPs in an Internet exchange, each serving subset of upstream ISPs, and (2) running the data and control plane of a TP on separate platforms. The first approach is easier to implement. The second approach offers the convenience of a single IP address for BGP sessions from ISPs and follows the best practices of data plane and control plane separation in modern routers. A data plane running on a separate platform could be implemented using OpenFlow or NetFPGA technologies.

Better routing stability in the face of transient client networks. The Transit Portal can support transient client networks that need BGP-based route control but do not need to use network resources all of the time. For example, suppose that a client network is instantiated every day for three hours to facilitate a video conference, or bulk transfer for backups. In these cases, the TP can simply leave the BGP prefix advertised to the global network, even when the client network is "swapped out". In this way, TPs could support transient client networks without introducing global routing instability.

Backhaul between sites. Today's cloud applications in different data centers, performing tasks such as backup or synchronization, must traverse the public Internet. For instance, Amazon EC2 platform offers sites in U.S. East coast, U.S. West coast and in Ireland. Unfortunately, the platform offers little transparency or flexibility for application operators seeking to connect the applications in different sites into a common network. TP platform, on the other hand, is well suited to support sophisticated backhaul between applications in different sites. Each TP site can choose among multiple paths to other TP sites and application operator could exercise control on what path applications are routed to reach other sites. In addition, TP could facilitate private networking between the applications in different sites by using tunnels between TPs. The TP could also improve connectivity between the sites through overlay routing, two TP sites exchange traffic through a third intermediate TP site.

2.7 Limitations

In this chapter we have presented and evaluated TP platform. The concept of TP and the evaluation are not without caveats. There are three distinct questions we haven't fully addressed in this chapter:

1. **Does TP provide the best way to communicate route choice to hosted services?** Transit Portal delegates all wide-area routing decisions to the hosted online services. While some services might benefit from such fine grained control, not all online services have resources and know-how to leverage available routes. What is the right level of wide-area route control is still an open research question. Some of the online services might benefit from coarser route choices. For example, a cloud provider might offer to its clients different route packages: value routes, low latency routes, and high throughput routes.
2. **How can a hosted online service choose among routes provided by TP?** When a hosted online service takes on to decide which wide-area routes to utilize, it has to be able to compare the performance of different routes. In one scenario, all of the services would evaluate the routes presented to them and make independent decisions. This scenario, however, poses significant overhead to each of the hosted online services. It is possible that in the future we will see cloud service providers providing information about wide-area route performance — akin to a network weather report. In such a setup, the measurements will be done by a could provider once and shared with the clients, thus eliminating the need for individual measurements from each client.
3. **How would upstream ISPs (and the Internet as a whole) react to an increase in routing churn?** Many industry practitioners believe that the current growth rate of the BGP routing table and the BGP update churn are unsustainable in long term. If cloud service providers addpt TP-like platform, it will likely increase both the

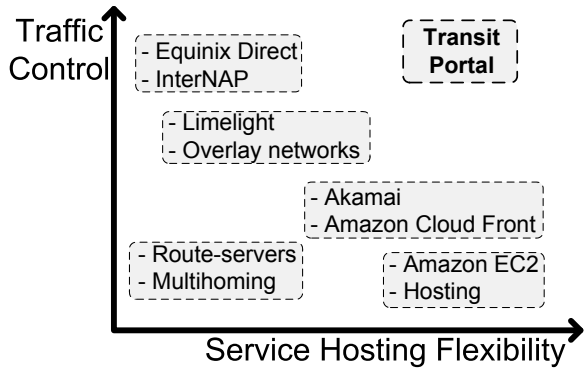


Figure 2.25: Transit Portals allow cloud providers to offer wide-area route control to hosted services

BGP routing table size and the update churn due to increased number of distinct participants in the Internet routing. On one hand, one could see that as a drawback of TP-like platform. On the other hand, this issue indicates that more research and industry efforts should be dedicated to improve the Internet scalability.

2.8 Related Work

The Transit Portal resembles several existing technologies, including content distribution networks, route servers, cloud hosting providers, and even exchange point operators. We describe how these existing technologies differ from TP, with respect to their support for the applications from Section 2.1.

Content distribution networks and cloud hosting providers do not provide control over inbound and outbound traffic. Content distribution networks like Akamai [15] and Amazon Cloud Front host content across a network of caching proxies, in an attempt to place content close to users to improve performance and save bandwidth. Each of these caching proxies may be located in some colocation facility with its own upstream connectivity. Some content providers may care more about throughput, others may care about low delay, others may care about reliability, and still others might care about minimizing costs. In a CDN, however, the content provider has no control over how traffic enters or

leaves these colocation facilities; it is essentially at the mercy of the decisions that the CDN provider makes about upstream connectivity. The Transit Portal, on the other hand, allows each content provider to control traffic independently.

Exchange points do not provide flexible hosting. Providers like Equinix Direct [43] allow services to change their upstream connectivity on short timescales and connect on demand with ISPs in an Internet exchange. Equinix Direct operates only at the control plane and expects clients and ISPs to be able to share a common LAN. Unlike Equinix Direct, the Transit Portal allows services to establish connectivity to transit ISPs without renting rack space in the exchange point, acquiring numbered resources, or procuring dedicated routing equipment.

Route servers do not allow each downstream client network to make independent routing decisions. Route servers reduce the number of sessions between the peers in an exchange point: instead of maintaining a clique of connections, peers connect to a central route server. Route servers aggregate the routes and select only the best route to a destination to each of the peers [54]. This function differs from the TP, which provides transparent access to all of the routes from upstream ISPs.

DNS-based load balancing cannot migrate live connections. Hosting providers sometimes use DNS-based load balancing to redirect clients to different servers—for example, a content distribution network (*e.g.*, Akamai [15]) or service host (*e.g.*, Google) can use DNS to re-map clients to machines hosting the same service but which have a different IP address. DNS-based load balancing, however, does not allow the service provider to migrate a long-running connection, and it requires the service provider to use low DNS TTLs, which may also introduce longer lookup times. The Transit Portal, on the other hand, can move a service by re-routing the IP prefix or IP address associated with that service, thereby allowing for longer DNS TTLs or connection migration.

Overlay networks do not allow direct control over inbound or outbound traffic, and may not scale. Some control over traffic might be possible with an overlay network (*e.g.*, RON [20], SOSR [56]). Unfortunately, overlay networks can only indirectly control traffic, and they require traffic to be sent through overlay nodes, which can result in longer paths.

2.9 Summary

This dissertation chapter has presented the design, implementation, evaluation, and deployment of the Transit Portal, which offers flexible wide-area route control to for distributed services. The TP prototype is operating at several geographically diverse locations with a /21 address block, AS number, and BGP sessions to upstream providers. We have used Transit Portal in both our research and in projects for a graduate course [83], and we plan to deploy and evaluate additional services, including offering our platform to other researchers, and to offer new, more lightweight interfaces to the Transit Portal.

CHAPTER III

QUANTIFYING THE BENEFITS OF WIDE-AREA ROUTE CONTROL

Online service providers (OSP) such as Google and Amazon are offering an increasingly diverse set of online services, from content streaming to interactive applications, including social networks and online productivity tools. Consumers expect these online services to be responsive; as a result, OSPs are continually implementing various optimizations to improve the performance of these services. Recent years have witnessed a plethora of optimizations to accelerate the delivery of online services, ranging from better transport protocols [4, 52] to browser enhancements [2, 64] to new compression algorithms [11] and site placement optimizations [31], to name a few. On shorter timescales, operators of online services continually manage client-perceived performance using algorithms for directing clients to different replicas (often termed *content routing*). In parallel network operators monitor the health of the current paths between the clients and the OSP replicas and, at longer timescales, perform various forms of traffic engineering to impact the *network routing* process. In particular, network backbone operators seek to optimize the performance of network paths between clients and individual server replicas.

The operational teams that manage content routing often have only limited coordination with teams adjusting the network routing [68]. The operators of major OSPs that we surveyed stated that their replica operators and network operators have only limited cooperation, and they do not attempt to reap the benefits of jointly optimizing content and network routing. Client performance suffers from this lack of coordination: operators of service replicas currently have no visibility into the performance or cost of alternate network paths between a service replica and its clients, so they optimize replica mapping

based upon the default network paths that have been exposed as a result of network operators' traffic engineering optimizations. On the other hand, network operators, who do have access to alternate wide-area paths, have little insight into the management of online services, or the application-level performance these alternate paths might provide.

Despite this technical and operational divide, past work on detour routing [98], overlay routing [20], and multi-homing [17, 18, 19] suggests that exploiting a range of alternate paths can offer significant performance gains over default wide-area Internet paths. These well-known results motivate us to explore jointly performing both content and network routing to improve the perceived client performance of online services. In particular, we consider whether the potential performance gains of both content and network routing can be combined in practice.

In this chapter, we design and evaluate PECAN (Performance Enhancements with Content And Network routing), a system that performs joint content and network routing for online services. PECAN enables joint content and network routing for online services by augmenting an OSP's existing content routing framework to provide a diverse set of wide-area routes between each replica and its clients. To ensure that PECAN does not harm the performance of any existing service, it explores alternate wide-area route choices using separate IP prefixes; clients can always reach the online service either via the default wide-area Internet routes or via the routes that PECAN exposes.

We evaluate a prototype deployment of PECAN to quantify the performance benefits joint content and network routing can achieve in practice. To deploy PECAN, we emulate an online service provider's infrastructure by placing replicas at the Transit Portal system (described in previous chapter of this dissertation) and clients at nodes in the PlanetLab testbed. TP allows us to emulate an OSP with a five geographically diverse, U.S.-based points-of-presence (PoPs), each of which provides access to many alternate wide-area paths to clients. There are many ways to measure a client's perceived performance; one popular metric is page load time, but accurately measuring page load time is challenging as it

requires instrumenting each client with browser software—a difficult task for a large-scale measurement study. Instead, we focus on network latency (i.e., round trip time), as many online service providers have identified latency reduction as a key factor governing an end user's experience [33, 34]. Moreover, our results indicate that latency is a reasonably good predictor for small-object download time in our deployment. We find that round-trip time and the time to download a median-size (400-byte [31]) Web object has a Pearson correlation coefficient of approximately 0.83. In addition to latency, we also report how PECAN improves both throughput and jitter.

Our results suggest that PECAN can provide significant performance benefits for today's online services. Using three months' of data from our globally distributed testbed, we show that, compared to content routing alone, PECAN improves the average client's latency to our online service by 22% more than content routing alone. Further, these gains are realizable in practice: exploring just five alternate routes at each replica can yield 60% of the average improvement possible when each client is free to choose among the hundreds of alternate routes that we tested in our experiments.

This chapter makes three contributions. First, we present the design and implementation of PECAN, a system for performing joint content and network routing in online service provider networks. Second, using millions of performance measurements over three months on a globally distributed testbed that emulates an online service provider network, we show that performing joint network and content routing can offer significant performance benefits to online services in practice. Although we focus on the resulting reductions in latency, we quantify PECAN's benefits for throughput and jitter as well. Finally, we decompose the performance results by studying how content routing and network routing alone reduce network latency on our testbed, which provides insight into *why* PECAN works, and provides confidence that our results are likely to apply to OSPs in general.

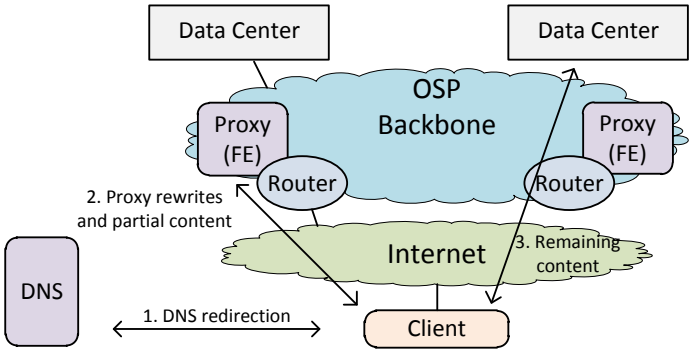


Figure 3.1: Global Traffic Management (GTM). Some services rely only on DNS; other services can use proxies for HTTP redirects and client-specific HTML rewriting.

3.1 State of the Art Overview

We begin by providing an overview of both content routing and network routing as employed by online service providers today. We attempt to describe the state of the art, as best as we can determine through discussions with network operators and online service providers. Because network and content routing typically occur independently in today's OSP networks, we describe each independently. PECAN effectively integrates both routing processes into one.

3.1.1 Global Traffic Management: Overview

Operators often refer to the system that controls how clients interact with their replicated online service as a *Global Traffic Management (GTM)* system. State-of-the-art GTM systems can perform both content and network routing, but independently. Content routing refers to the process of selecting which data center replica (among a set of geographically distributed points-of-presence) should service a particular request. Network routing, in contrast, refers to the process of selecting both the wide-area, interdomain paths and the intra-domain paths within an OSP backbone that each replica will use to communicate with remote clients.

Figure 3.1 shows an example GTM system which has several components involved in directing traffic. The example shows a system that can perform content routing with both

DNS-based direction and proxy-based redirection (*e.g.*, with URL rewriting). In this dissertation chapter, we do not distinguish between the different ways that a network operator might perform content routing; we merely assume that an OSP can map clients to replicas in at least one of these ways. If the OSP operates its own backbone network with interdomain routing connections to the wide-area Internet, it can also perform network routing to adjust the paths that client traffic takes to reach each replica. For example, An OSP operator might perform network routing by adjusting intra-domain routes on the OSP backbone, or by performing interdomain traffic engineering at the border routers between the OSP backbone and the OSP's peers.

Despite the flexibility that these GTM systems provide, OSPs may still have trouble achieving good performance for their clients, since content routing and network routing remain disjoint. On the one hand, content routing systems have limited visibility into (and control over) the alternate paths that are available to network routing systems. On the other, network routing systems often have relatively poor visibility into the end-to-end performance that clients experience and how changes in network routing could improve performance. The goal of PECAN is to bridge this gap.

3.1.2 Content Routing

Content routing systems have been heavily influenced by academic research, which we overview in Section 3.6. Today's content routing systems perform three major functions: (1) collecting performance information, (2) mapping clients to replicas, and (3) directing clients to replicas according to the client-replica map. We describe these steps below.

Step 1: Collecting performance information. OSPs use many technologies to measure performance between online service replicas and their clients. Such measurements can be classified into *active*, *passive*, and *indirect*. Active measurements usually involve sending probes from replicas to clients (or *v.v.*), which provides direct information about the network performance to the client. Active measurements are problematic in at least two

ways: (1) the probes might not be handled by network in the same way the actual traffic is handled, and (2) active measurements do not scale to a large number of replicas and customers [60]. For these reasons, in practice OSPs typically use a combination of passive and indirect measurements.

Passive measurements record the performance of the actual online service traffic between the replica and the clients to estimate the performance that clients are receiving. Passive measurements scale well and can provide direct insight into the performance of a service over the network, recording information such as TCP round-trip times and packet loss. To measure performance for all <client, replica> pairs, OSPs occasionally randomly redirect a small fraction of their clients' requests to alternate replicas, as suggested by Seshan *et al.* [100].

Step 2: Mapping clients to replicas. OSPs use a variety of proprietary algorithms to map clients to replicas. Client-to-replica performance is important, but there are other inputs to the algorithms that produce these mappings as well, including service availability, servicing costs, desired load, and regulatory restrictions [114].

To map clients to replicas when no active or passive measurements are available to inform selection, OSPs often resort to indirect inference of the likely performance between replicas and clients. Commercial IP geo-location databases [75] are often augmented with historical information to estimate performance between replicas and clients. As new clients begin to use the system, the OSP can update its performance estimates with passive measurements.

Step 3: Directing clients to replicas. OSPs use three main techniques to implement their client-to-replica mapping: (1) DNS-based redirection, 2) HTTP redirection, and (3) client-tailored HTML rewriting. *DNS mapping* uses DNS servers to respond to clients with IP addresses of best replicas. ("Clients" in this case most often are the DNS resolvers resolving names on behalf of the end-systems.) DNS mapping is most useful to improve

the performance of initial resource requests. *HTTP redirection* can further redirect clients to a better replica (at a cost of initial request latency.) When the requested resource has multiple sub-components (*e.g.*, a Web page with images), the OSP can use *client-tailored HTML rewriting* to direct clients to retrieve sub-components from disparate replicas.

Taken together, these methods allow online service operators to achieve good client performance while spreading load across their infrastructure. Yet, these content routing mechanisms operate only on the paths that are already chosen by the network operators. Next, we describe a set of popular methods that network operators use to select the paths between clients and replicas.

3.1.3 Network Routing

Industry has taken two main approaches towards network routing: (1) deploying commercial platforms and services for multi-homed enterprises; and (2) performing in-house traffic engineering by adjusting network configurations in the OSP backbone network. We discuss each of these approaches below.

Commercial platforms and services. Avaya and Cisco offer the PathControl [97] and Optimized Edge Routing (OER) [37] route management platforms, respectively. These platforms perform continual performance measurements to online services and adjust interdomain routes between the services and their clients based on these performance measurements. Similarly, Internap provides route optimization services for their clients [62] by performing measurements along alternate paths and redirecting traffic between services and clients by adjusting interdomain routing policy. These platforms and services primarily target large enterprises with multiple upstream providers. PECAN applies similar types of interdomain route control to adjust routes between clients and replicas, and it is possible that some variant of these systems could be used to implement aspects of PECAN's network routing subsystem. Our evaluation also hints at how these services might scale to large OSPs who have have millions of clients and many replicas.

Conventional traffic engineering. Large OSPs primarily improve the performance of their connectivity by deploying their own networks and increasing the richness of their peering connections to improve wide-area network performance. Peering significantly reduces transit costs for large ISPs, but it can sometimes harm performance. For example, Zhang *et al.* showed that by preferring peering connections, Microsoft's users on average would experience an RTT of 66 ms—more than twice the default routing policy which gives an average RTT of 29 ms [117]. This anomaly indicates that peering alone is not always effective as performance enhancing strategy.

Despite the benefits of Internet route control as highlighted in many research studies (see Section 3.6), the network operators that we interviewed at large OSPs rarely use anything more sophisticated than general routing policies (*e.g.*, setting per-link BGP `localpref` parameters or export policies). In 2003, Feamster *et al.* found that wide-area routing heuristics used in ISP backbones were primitive and *ad hoc* [47], eschewing available granular routing techniques like per-prefix `localpref` or export policies. Our conversations indicate that the state of affairs is largely the same today: While network operators do occasionally use granular routing, they do it only in exceptional cases, when service degradation is noticeable and content routing alone cannot provide satisfactory resolution. For example, the "WhyHigh?" system proposed by Krishnan *et al.* [68] (and deployed in Google) identifies corner cases of prefixes exhibiting higher delay than expected. Systems like "WhyHigh?" can alleviate occasional problems with network routing, but they are reactive. In contrast, PECAN aims to systematically and proactively look for best performing network paths and optimize them jointly with content routing.

3.2 Case for Joint Content & Network Routing

To quantify potential gains of a joint content and network routing we emulate an OSP setup using a globally distributed testbed that allows us both to replicate services across sites and control inbound routes to these sites. This testbed, which we describe in detail

Table 3.1: Average improvement to latency (RTT), throughput (BW), and jitter. The baseline, over which improvement is measured for each technique, is the average performance to the single best replica.

Routing type	RTT	BW	Jitter
Best replica	107.35 ms	212.47 Mbps	5.95 ms
Network	4.35%	0.87%	9.32%
Content	16.75%	8.11%	11.82%
Joint	**20.44%**	**11.29%**	**17.57%**

in Section 3.4, allowed us to emulate an OSP with replicas in a diversity of geographic locations and a diversity of wide-area network routes at each location.

Overview of measurements. The testbed has five replicas distributed across the United States; from each replica, we explore about 250 alternate routing choices to about 200 globally distributed clients. For six months (from July to December 2011), we collected a comprehensive <client, replica, route> performance map consisting of millions of measurements. We have released this dataset to the reviewers at an anonymous site [3] and we plan to make that data set public with publication of this work. Section 3.4 explains our experimental setup in more detail.

Improvement over the best replica. When OSPs roll out a new online service, it often starts at a single replica and then expands to more sites. It is interesting to know how expanding the set of replicas and/or adding joint content and network routing improves the service performance. Table 3.1 shows how network routing, content routing and joint routing improves over a single best replica.

The "network" routing row in Table 3.1 shows improvement gains if the OSP chooses to explore alternate routes only for that single best replica. Conversely, the "content" routing line shows improvement gains if the OSP chooses to replicate the service to all five locations available in our testbed. Content routing provides greater performance gains than simply applying network routing for one site. Finally, the "joint" routing row shows the gains attained when the OSP chooses to both replicate the service to all five locations and

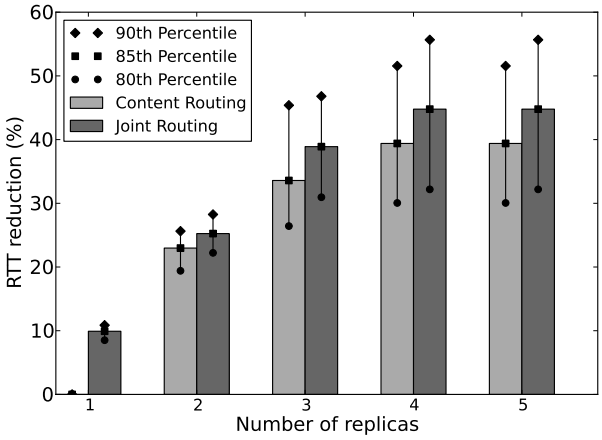


Figure 3.2: Latency reduction as a function of the number of replicas.

Table 3.2: Marginal gains of joint routing over content routing. Clients in each percentile improve by at least the amount indicated.

Client percentile	RTT (%)	BW (%)	Jitter (%)
Most-improved 5%	16.94	10.25	41.20
Most-improved 10%	13.43	1.42	29.24
Most-improved 20%	9.91	0.47	13.48

perform joint content and network routing.

In practice, in addition to network level performance, both replica and network path selection depend on traffic acquisition costs, replica loads, and other variables. Unfortunately, it is hard to obtain data to model the effect of such variables. Hence, we optimize only for latency, throughput, and jitter. While this might bias our results, it would affect both content routing and joint routing and thus we can still compare the two.

Figure 3.2 shows that the benefits from joint routing are largely independent of the size of the replica set in our testbed: adding more replicas to an OSP yields latency improvements for both content and joint routing. Hence, an OSP can improve its performance using joint routing regardless of the number of replicas it currently employs. The numbers in the figure show the 80th, 90th and 95th percentile gains over the performance of a single best replica (as defined in Section 3.5.)

Improvement over content routing. While joint routing unquestionably provides greater gains than network or content routing alone, most OSPs already deploy content routing. Hence, its practical impact is governed by the marginal gain over content routing. Table 3.2 provides a breakdown of joint routing performance gains over content routing alone for various client percentiles. For example, when compared to content routing, the most-improved 10% of clients see their latency reduced by 13.43% or more, an increase of 1.42% or greater in throughput, and at least a 29.24% reduction in jitter. For online services such as search or online gaming, such latency savings are significant.

Improvement with limited route diversity. It may not be practical to support all possible alternate routes to each replica. Fortunately, Figure 3.15 (Section 3.5) shows that exploiting just five alternate routes for each of the replicas can yield 60% of the possible improvement across the hundreds of alternate routes that we tested in our experiments.

Taken together, these results suggest that an OSP can provide a tangible improvement over the state of the art by employing joint content and network routing. In the next section, we will describe the design of PECAN, a joint route and replica selection system that can realize these gains in practice.

3.3 PECAN: Performance Enhancements with Content and Network Routing

As we described in Section 3.1, modern OSPs use sophisticated content routing systems to load balance requests between replicated data centers in an attempt to improve client performance. In this section, we will describe how operators can extend their existing content routing systems to support network routing as well. In particular, we present the design of PECAN (Performance Enhancements with Content And Network routing), a system that enables seamless integration of content and network routing.

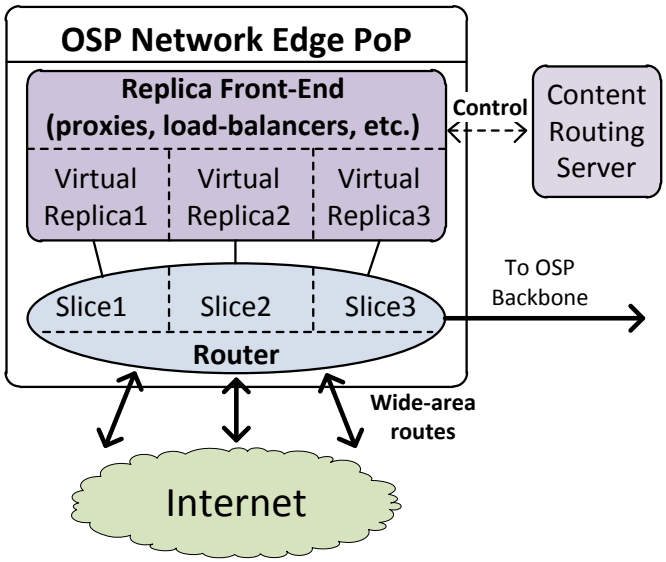


Figure 3.3: Content and network routing subsystems have to allocate isolated resources for exploration of new network routes.

3.3.1 Exposing Routing Choices

The basic idea behind PECAN is to extend an OSP's current <client, replica> mapping infrastructure to instead map clients to <replica, route> pairs. To do so, PECAN breaks each replica into a set of virtual replicas, where each virtual replica corresponds to a different choice of routes to the replica (i.e., a single <replica, route> tuple). Figure 3.3 shows how PECAN allows a content routing system to tap into network route diversity. The router in the figure has a separate routing slice dedicated to each set of alternate routes (virtual replica). For example, in today's routers such a slice can be implemented using Virtual Routing and Forwarding instances (VRF), but a variety of alternative technologies can be employed as we discuss below.

3.3.1.1 Egress routes

There are a wide variety of mechanisms available to employ alternate egress routes from a given virtual replica. For example, conventional BGP multihoming can increase route diversity; operators can use BGP's local preference setting to adjust the choice of egress routes to each client prefix. PECAN could also benefit from protocols such as Detour [98],

RON [20], Platypus [94], Deflections [116], and Path Splicing [80], all of which increase an end-systems choice of (and control over) egress paths. Similarly, considering industry proposals, many practitioners see Locator/ID Separation Protocol (LISP) [45] as a feasible improvement to BGP. LISP separates the Endpoint Identifier (EID) (*i.e.*, a host IP address) information from Routing Locator (RLOC) (*i.e.*, the information that encodes the location of the EID in the wide-area Internet.) LISP could allow an OSP to explore potential egress routes by selecting the entry points to a remote network as encoded with RLOCs.

3.3.1.2 Ingress routes

Affecting a virtual replica's ingress routes is more challenging since it requires changing the way other networks forward packets. The key to enabling distinct route sets for each virtual replica is to separate these routing decisions. In PECAN, an OSP allocates a distinct IP address prefix to each virtual replica. Hence, to map clients to a particular virtual replica, PECAN need only point them to an IP address within the virtual replica's prefix.

Today's Internet supports a number of ways to impact route selection, including selective prefix announcement (*i.e.*, announcing a prefix only to a subset of neighbors), prepending `AS_PATH` attributes, setting BGP communities or MED attributes, and BGP `AS_PATH` poisoning; we evaluate employing AS path poisoning in PECAN extensively in the next section. Future technologies, such as LISP, might provide even more elegant alternatives.

Critically, by maintaining one virtual replica (address prefix) at each physical replica that always uses the default network paths, PECAN ensures that it can do no harm: clients can always obtain the performance provided by content routing alone if none of the joint routing options provide superior performance.

3.3.2 Selecting a Virtual Replica

We now describe how PECAN's virtual replicas enable the process of joint content and network routing. The joint optimization could happen in many ways; we take an iterative approach, as shown in in Figure 3.4. First, PECAN optimizes network routing between

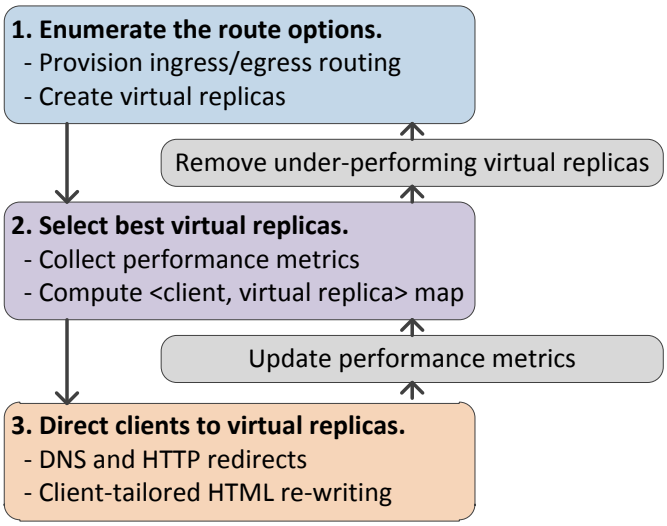


Figure 3.4: Joint network and content routing selection with PECAN.

each client/replica pair: for each replica, PECAN identifies the network path to each client that yields the best performance, and establishes a virtual replica with that path preference. Then, for each client, PECAN selects the virtual replica that offers the best performance among the available options at each physical replica.

This process proceeds in three steps, which could be either automated or manually performed by the operators of the network and online service.

1. **Enumerate the route options.** OSP network operators must enumerate the alternate routes from each replica to clients that the system should explore. Depending on the route selection technique employed, the operator may wish to enumerate egress (*e.g.*, a choice of a next-hop neighbor) routes and ingress (*e.g.*, selective route announcement) routes separately, or jointly. Evaluating all possible alternate routes to each client is unlikely to scale, but our evaluation (Section 3.5) shows considering just five virtual replicas (i.e., sets of alternate ingress routes) at each replica can realize performance improvements that are 60% of the maximum possible improvement.
2. **Select the best virtual replica for each client.** PECAN evaluates the performance of each virtual replica for each client. To evaluate a new virtual replica (route selection) for a given client and physical replica, PECAN redirects a small fraction of client

Table 3.3: The Transit Portal deployments that we use to emulate a replicated online service with route control. At each Transit Portal location, we host a replica of an online service; from each of these locations, we explore more than two hundred alternate routes between each replica and the set of clients that it could reach.

Service Replica	Location	# of routes	# of measurements (RTT)		# of measurements (BW)	
			Default route	Alternate routes	Default route	Alternate routes
1	Atlanta, GA	259	292,806	1,453,137	9,535	14,679
2	Clemson, SC	253	19,401	1,442,832	14,853	22,021
3	Princeton, NJ	261	224,457	1,438,588	5,595	6,243
4	Seattle, WA	247	366,357	347,302	14,844	9,651
5	Wisconsin, WI	247	67,473	1,389,266	7,321	14,032

requests to the virtual replicas and evaluates the performance that the client sees. PECAN gradually increases the number of clients mapped to a virtual replica to avoid overloading any network path or physical replica. Isolating test measurements from the bulk of the traffic requires a set of dedicated load-balancing proxies, as shown in Figure 3.3, As long as the evaluated route offers improved performance for enough clients and is reliable over the test period, PECAN maintains the virtual replica in the set of virtual replicas that can be used for joint routing.

3. **Direct clients to virtual replicas.** Once PECAN has selected the best virtual replica for each client, it implements the mapping. To implement this mapping, PECAN uses DNS load balancing to map each client to a virtual replica IP address, where the BGP prefix for that IP address corresponds to the route that the PECAN has selected for that client and replica using the previous steps. Because PECAN maps each virtual replica to its own prefix, a client always has the option of using either default content routing (*i.e.*, the route it would receive in today's CDN) or the PECAN-provided route.

3.4 Evaluation Setup

This section describes our evaluation methodology. We first describe the testbed infrastructure, followed by our measurement procedure.

3.4.1 Infrastructure

We use PlanetLab to emulate a set of clients, from which we perform measurements to the replicas over many different sets of routes, and the Transit Portal to attain both replica and route diversity.

3.4.1.1 Clients: PlanetLab

We use 200 PlanetLab nodes as our client set. From the full list of PlanetLab nodes, we select nodes with which we can establish sessions. We further filter the list of these "alive" nodes to include only a single node per PlanetLab site. In the end, we have a client pool with 38% of the nodes in North America, 36% in Europe, 21% in Asia, and the last 5% in South America.

It is well known that PlanetLab nodes are not the best representation of the Internet. It is hard to quantify how much PlanetLab biases our measurements. On one hand, PlanetLab nodes are better provisioned and have better "last mile" connection to the immediate provider than their residential counterparts. On the other hand, we focus our measurements on the performance we can gain by exploiting replica and route diversity in network core and not on the network edge. In other words, what matters more is how well a PlanetLab node's provider is connected to the Internet as compared to an average Internet user. Most PlanetLab node's access providers are academic institutions, whose connectivity to the Internet is often comparable to the connectivity of smaller ISPs or medium enterprises.

3.4.1.2 Replicas: Transit Portal

Transit Portal (TP) is a research platform for realistic experimentation with wide-area route control, which we described in previous chapter of this dissertation. TP consists of several nodes, each of which is a functional Internet router, connecting to an upstream ISP, receiving a full Internet routing table and able to participate in BGP routing by issuing BGP updates from the IP address space and AS numbers allocated to Transit Portal. TP nodes

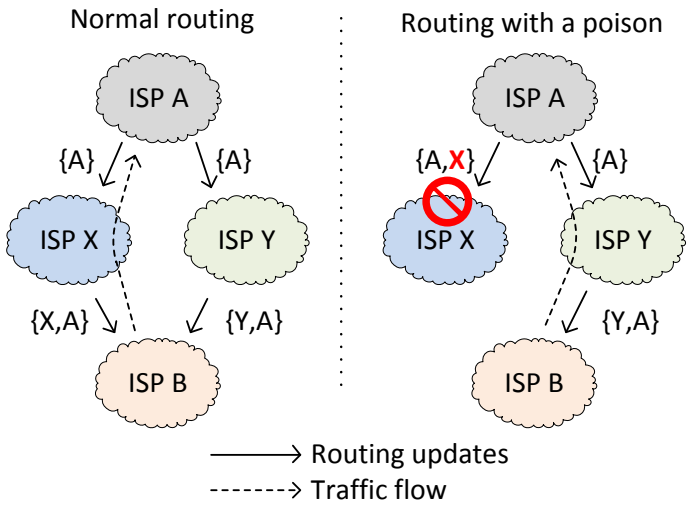


Figure 3.5: Obtaining alternate paths between ISP A and ISP B with poisoning.

allow multiple researchers to use these routing resources concurrently. We used access to the five TP nodes described in Table 3.3; each of which acts as a replica in our testbed.

3.4.2 Experiment Procedure

We describe how we use our testbed to explore alternate routes between clients and replicas measure the performance improvements that result from these alternate routes.

3.4.2.1 Obtaining route diversity: poisoning

We explore wide-area route diversity by using BGP `AS_PATH` poisoning [30]. BGP `AS_PATH` poisoning is an unconventional technique: it finds alternate, policy-compliant wide-area ingress routes to the network that advertises the poisoned route (in our case, the replicas). In practice, a real OSP could control both ingress and egress routes in a variety of ways. For example, OSPs could also use BGP `AS_PATH` prepending, BGP community attributes, and selective advertisements; to control egress traffic, OSPs can often select among multiple neighbors to send traffic. Among these options, BGP `AS_PATH` poisoning is the only feasible choice to obtain wide-area route diversity because we do not have access to the BGP routers that control inbound and outbound traffic to our replica sites. Given a wider variety of techniques at their disposal, it is plausible that OSP operators might see even

greater performance gains than our experiments reflect.

BGP AS_PATH poisoning leverages the BGP-loop prevention algorithm to explore alternate routes on the Internet. Each router in the Internet attaches its own AS number to AS_PATH attribute. BGP's loop prevention algorithm, which is implemented on all BGP-speaking routers, says that a router must drop a BGP route update if the AS_PATH attribute of the route contains the AS number of said router. Dropping these updates prevents the router from accepting updates that the router has already received, thus preventing loops. As shown in Figure 3.5, BGP AS_PATH poisoning exploits this algorithm by inserting a target AS number in the AS_PATH attribute before the update is originated. The target AS, in turn, will drop the update and its clients will choose alternate routes to the route originator.

We use the traceroute tool to identify the networks (and their AS numbers) on the default paths from our clients to each replica. We then poison these AS numbers one by one to uncover alternate paths from clients to replicas. Not every client moves to an alternate path after we issue a poisoned update: some updates affect just a few clients, while some affect a great many. To determine how often the AS path changes as a result of a poisoned update, we traceroute the resulting path to the updated IP prefix. When measured from an average client, approximately 20% of all poisoned announcements produce paths different from a default. On average, we find 3.4 different AS paths per <client, replica> pair (not counting the default path) in our testbed. There is a possibility that a small fraction of these alternate paths are because of network topology changes not under our control—i.e., not because of our poisoning. We perform extensive measurements over the default paths to see how often the AS path changes, or in other words, to determine the "noise floor" of the Internet topology churn. We find that an average <client, replica> pair observes approximately 0.35 paths in addition to the default path during our study period of 3 months. This low churn estimate gives us confidence that most of 3.4 alternate paths (not counting the default) are indeed due to poisoning.

3.4.2.2 *Measuring performance improvements*

There are many ways to measure network performance improvement. We do not measure page load times due to the complexity of such measurements. Instead, we considered measuring median and average HTTP object download times, but find that latency is a reliable predictor for such times. During preliminary testing we discovered that round-trip-time correlates to download time for a median-sized (400-byte [31]) object with a Pearson correlation coefficient of approximately 0.83, for both a default path as well as a poisoned path that affected 30% of the replica clients. Instead, we measure three basic network performance primitives: latency, throughput, and jitter.

Each of our replicas maintains an un-poisoned prefix announcement at all times. This un-poisoned prefix can always be reached to perform a measurement over the default path that rarely changes. For poisoned routes, we use the following sequence for each replica to collect a client/replica path performance map:

1. **Announce a prefix with a poisoned update.** The poison will propagate the prefix to some client networks over the alternate paths.
2. **Perform measurements to the poisoned prefix.** From every client in our client set, collect measurements to the replica using the poisoned prefix. Clients for which poison did not affect the end-to-end path will see no improvement. Clients for which the prefix affected the end-to-end path will see either improved or reduced performance.
3. **Perform measurements to an un-poisoned prefix.** Conduct the same set of measurements over the default path (*i.e.*, using the un-poisoned prefix) to the replica to collect a contemporaneous baseline to which we will compare our poisoned path.

As shown in Table 3.3, the dataset resulting from these measurements has many more measurements over the default path than over poisoned paths. (Note the number of measurements reported for each metric in Table 3.3 corresponds to the total over all alternate routes; the average alternate route has roughly 250$\times$ fewer measurements.) The abundance

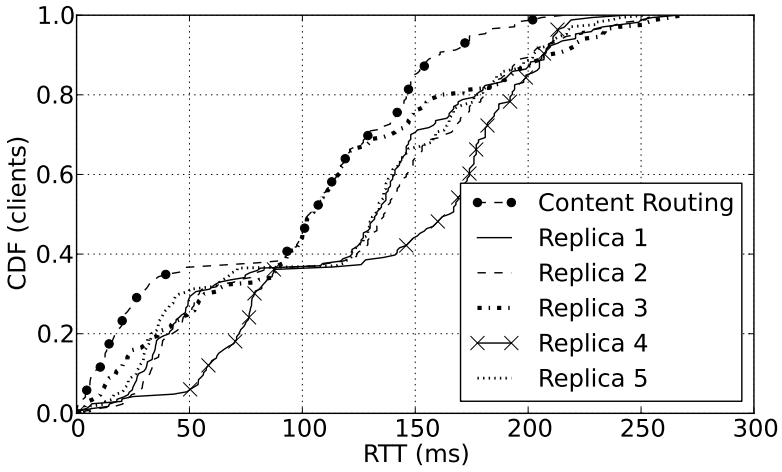


Figure 3.6: Minimum latency over the default path.

of default path measurements allows us to set a firm baseline. We consider a poisoned path between a client and a replica to improve latency over the client's default path only if the poisoned path shows latency smaller than the minimum latency ever recorded over the default path between the client and the replica. We apply a similar litmus test for jitter measurements. For throughput, we record an improvement only if a poisoned path produced higher throughput than any throughput measurement we ever observe on a default path.

3.5 Evaluation

We evaluate the benefits of joint routing with respect to latency, throughput and jitter. We also compare how well joint routing performs compared to traditional content routing.

3.5.1 Baseline

When considering the performance improvements that different routing approaches induce, we must establish a baseline to compare them against. In this section, we use two baselines for comparison: 1) a *best replica baseline*, and 2) a *content routing baseline.* Before defining these baseline metrics in detail, we describe the measurements we perform.

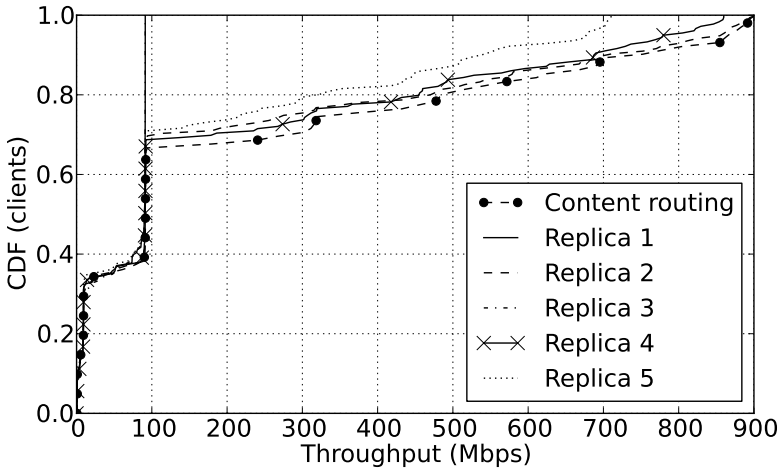


Figure 3.7: Maximum throughput over the default path.

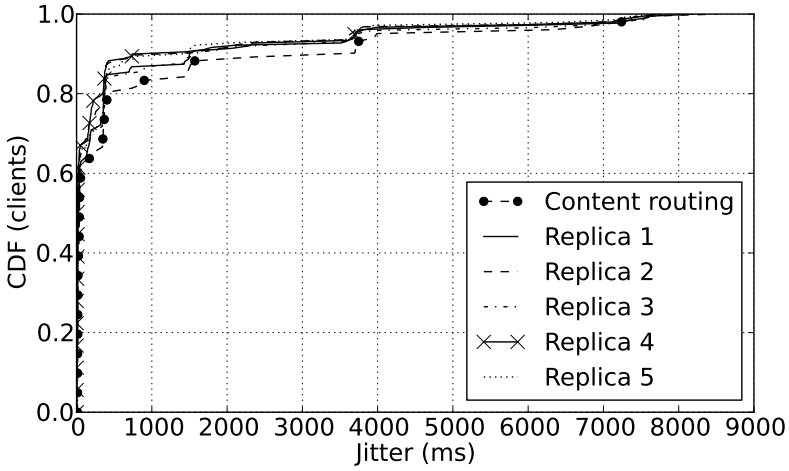


Figure 3.8: Minimum jitter over the default path.

Measurements. Figure 3.6 shows the CDF of *minimum latencies* clients experience to each replica over a default path. The minimum latency for each client is obtained from a large set of measurements: On average, each client measures the default path to a replica 6,692 times. The figure shows two major groups of clients: About 40% of the clients have latencies between 0–50 ms, and about 60% of the clients see latencies of 90 ms or larger, with just a few in between. These modalities reflect the geographic distribution of our client dataset: About 38% of clients are in the U.S and Canada and see lower latencies, while the rest of the clients are overseas.

Table 3.4: Percentage of clients for which a replica is the best choice.

Replica	Latency	Throughput	Jitter
1	6.93%	7.77%	12.62%
2	1.15%	35.92%	35.92%
3	56.64%	6.77%	8.74%
4	22.54%	48.54%	18.45%
5	12.71%	0.97%	24.27%

Similarly, Figures 3.7 and 3.8 show the CDFs of *maximum throughput* and the *minimum jitter*, respectively, as observed by clients to each replica. As with latencies, the maximums and minimums are computed over a set of measurements for the default path between each <client, replica> tuple. For each such tuple, we have, on average, 189 jitter and throughput measurements. Figure 3.7 highlights why it is difficult to measure capacity using the PlanetLab nodes as clients. The clients in the figure form three distinct groups: 1) those with 10 Mbps links, 2) those with 100 Mbps links, and 3) those with speeds above 100Mbps. The 10 Mbps and 100 Mbps groups identify cases where the PlanetLab nodes are directly connected to a bottleneck link; in the first two cases the bottleneck is at the client itself and neither joint routing nor network routing can improve throughput.

Best replica baseline. When a new online service is launched, it often starts with a single replica. We want to know how much the network performance improves over that single replica when the OSPs start adding more replicas and implement content routing or joint routing. We define the *best replica* for some performance metric as the replica that the largest fraction of clients would select, given that each client can select its own best replica based on that performance metric. Table 3.4 shows the breakdown of popularity of different replicas when each client selects a replica based on the performance of the default paths for each <client, replica> tuple. The average performance to the best replica across all clients yields the *average best replica performance*. For example, as shown in Table 3.1, the average latency that clients experience to the best replica (which, from Table 3.4, is replica 3) is 107.35 milliseconds.

Content routing baseline. In some ways, PECAN extends content routing to also incorporate the benefits of network routing; to quantify the additional benefit of PECAN relative to content routing, we also compare the performance of PECAN against the performance that content routing provides. Recall that, a content routing system maps each client to its own best replica. Formally, for a client i, the content routing latency $RTT_{content,i}$ is:

$$RTT_{content,i} = \min_{j \in (1...M)} \widehat{RTT}_{ij0} \tag{3.1}$$

where M is number of replicas and $\widehat{RTT}_{ij0}$ is the minimum latency that we measured between client i and replica j over the default path (noted as path 0). Taking the average across all clients yields *average content routing performance*. In addition to showing the performance of clients to each replica Figures 3.6–3.8 also show the performance of a content routing system when each client is directed to its own best replica. Note that the content routing system we implement accounts only for the performance data; in practice, OSPs also use replica loads and costs to perform content routing, but we do not have access to such metrics.

3.5.2 How Well Does Joint Routing Work?

We quantify the benefit that PECAN provides when compared to content routing. When PECAN is in use, we formally define client's i latency as:

$$RTT_{PECAN,i} = \min_{j \in (1...M)} \min_{l \in (0...K_{ij})} \widehat{RTT}_{ijl}, \tag{3.2}$$

M is the number of replicas, K_{ij} is the number of paths between client i and replica j, and $\widehat{RTT}_{ijl}$ is the minimum latency we recorded over the path l between client i and replica j. Path $j = 0$ represents the default path. With measurements of both content routing and PECAN performance, we can compute the percentage improvement as $100(RTT_{content,i} - RTT_{PECAN,i})/RTT_{content,i}$. We use the same approach for jitter; for throughput, we take maximums instead of minimums. Below we provide a breakdown of the average performance improvements that we presented in Table 3.1 from Section 3.2.

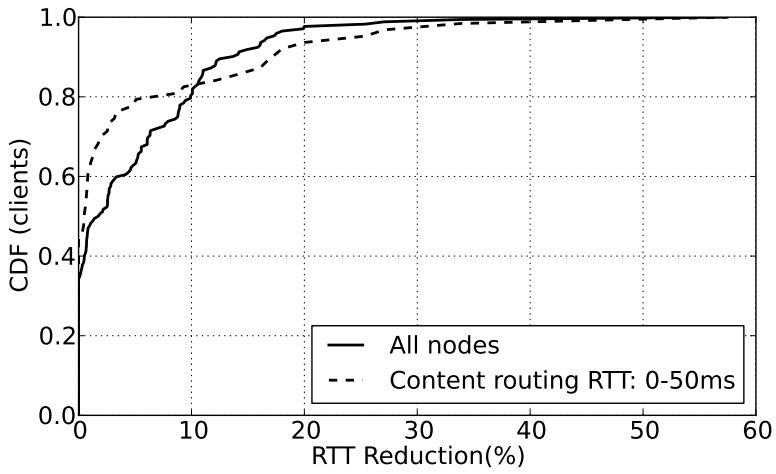
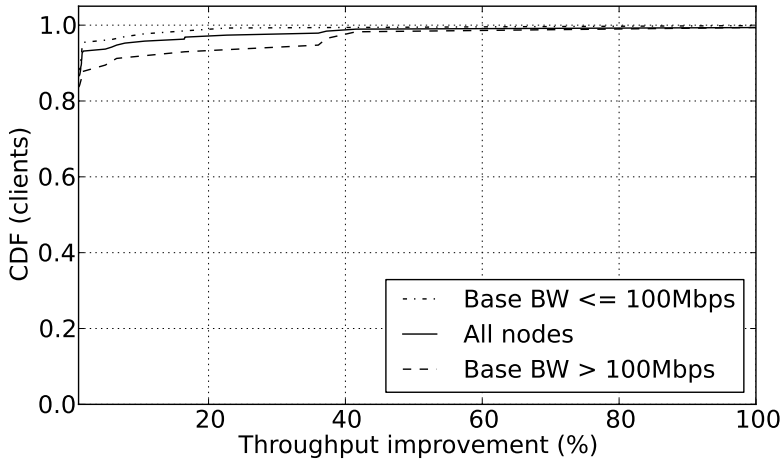


Figure 3.9: Latency reduction when using joint routing.

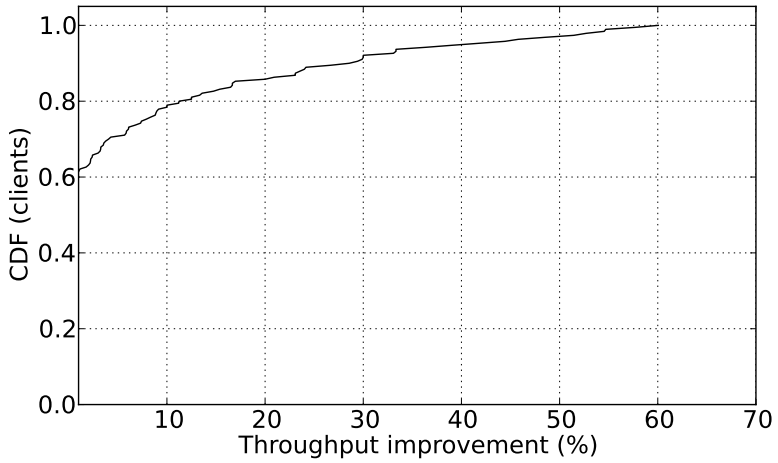
Latency. Figure 3.9 shows the percentage improvement in latency over the *content routing baseline*. The solid line in the figure shows improvement for all the clients, while the dashed line shows shows latency reduction for clients that had a baseline latency of 0–50 ms. We find that 20% of our clients see a reduction in latency of at least 10%. We also observe that clients with baseline latencies of 0–50 ms see similar improvements, with 82% of these clients improving by 10% or more. This result is significant, since content replication can only reduce latency by placing content closer to the users. At some point however, placing replicas close to all clients might become prohibitively expensive; in such cases, an OSP might rely on PECAN to improve routing between the closest replica and the clients nearby.

Figure 3.2 plots content and joint routing benefit over the *best replica baseline* as we increase the number of replicas. Adding replicas provides higher improvements with joint routing, but with decreasing marginal improvement at every addition. Content routing behaves similarly, albeit with a lower improvement at each step.

Throughput. Figure 10(a) shows the throughput gains that result from joint routing. When compared to the baseline of content routing, about 5% of clients that use joint routing experience performance improvements of 20% or more. As expected, the improvement



(a) Throughput.

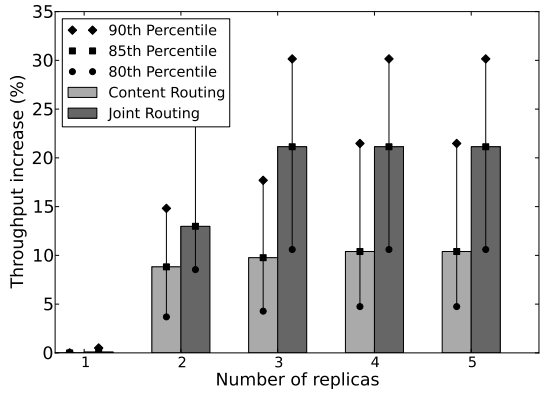


(b) Jitter.

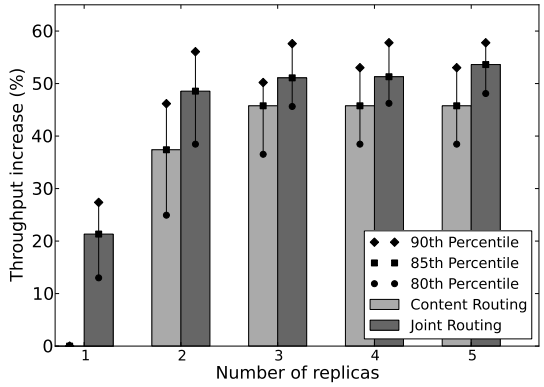
Figure 3.10: Joint routing vs. the content routing baseline.

is greater for clients whose baseline throughput values are greater than 100 Mbps. For such clients, it is more likely that the bottleneck is in the wide-area network.

Figure 11(a) shows how both content routing and joint routing improve performance as we increase the number of replicas. We see that the replica with the best throughput for most clients sees little improvement from joint routing. When we add the second best replica, the 85th percentile performance of content routing and joint routing increases to 8.5% and 13%, respectively. Adding more than three replicas, however, provides no additional gains to either content or joint routing. This, again, highlights the limitation of the



(a) Throughput



(b) Jitter.

Figure 3.11: Joint and content routing vs. the best replica baseline when increasing the number of replicas.

throughput measurements from the network edge: in most cases either the end clients or the replicas themselves are the bottleneck, so changing network paths or adding replicas with low-speed interfaces will not improve performance.

Jitter. Figure 10(b) shows how joint routing can reduce jitter, as compared to the baseline of content routing. For 10% of the clients, joint routing provides gains by approximately 30%. Figure 11(b) shows how both content and joint routing reduce jitter as we increase number of replicas. Jitter gains with increasing number of replicas exhibit similar patterns of marginally decreasing gains (Figure 3.2).

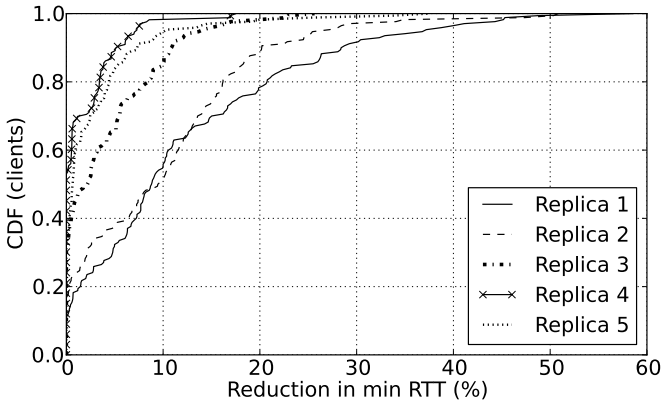


Figure 3.12: Latency reduction with network routing.

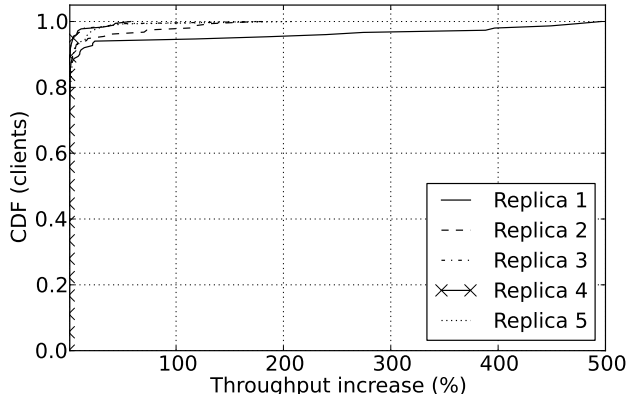


Figure 3.13: Throughput increase with network routing.

3.5.3 Why Does Joint Routing Work?

Joint routing improves performance over content routing because it provides multiple alternate network paths for each replica. As explained in Section 3.4, PECAN finds about 3.4 alternate paths on average for each <client, replica> tuple. Even when a client cannot improve its performance by switching replicas, network routing can often improve performance to one of the replicas. Figure 3.12 shows latency improvements from network routing alone for each replica and its clients. For each replica, the baseline over which we compute the latency reduction is the minimum latency over the default path to that replica. We find that 20% of the clients experience improvements of 5–20%, depending on which replica we choose to evaluate.

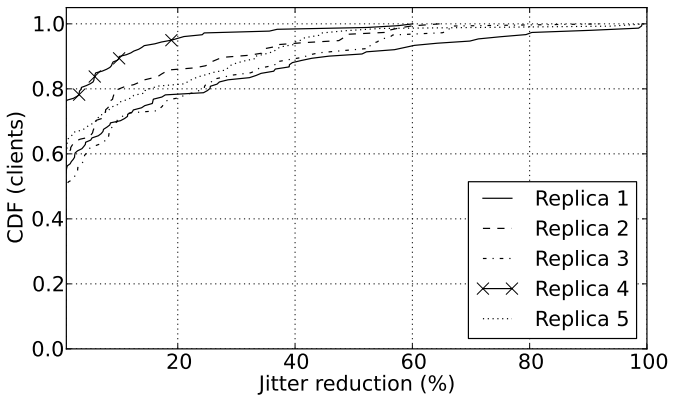


Figure 3.14: Jitter reduction with network routing.

Similarly, Figure 3.13 shows how network routing can improve the throughput between a client and its assigned replica. With our setup, where in many cases throughput has a bottleneck in the network edge, we find only between 10 to 20% of the nodes see throughput improvements at all. When network routing does achieve an improvement, however, it can improve throughput by as much as a factor of five. Finally, Figure 3.14 shows the reduction in jitter when replicas apply network routing; we find that most of the replicas can find paths with lower jitter for about 40% of clients.

3.5.4 Scalability and Stability

To be practical, PECAN must judiciously limit the number of route changes it broadcasts to the Internet; it also should limit oscillations by causing large numbers of clients to change replicas. In this section we seek to assess these requirements by analyzing two questions: 1) How many routes must the OSP explore to achieve the benefits of joint routing?; and 2) How many clients change their preferred replica after joint routing is applied?

3.5.4.1 Scaling route selection

How many routes must the OSP explore to achieve the benefits of joint routing? OSPs that tweak egress routes can do so without affecting the Internet routing system. To explore the alternate ingress routes, the OSPs have to issue additional routing updates. To this point

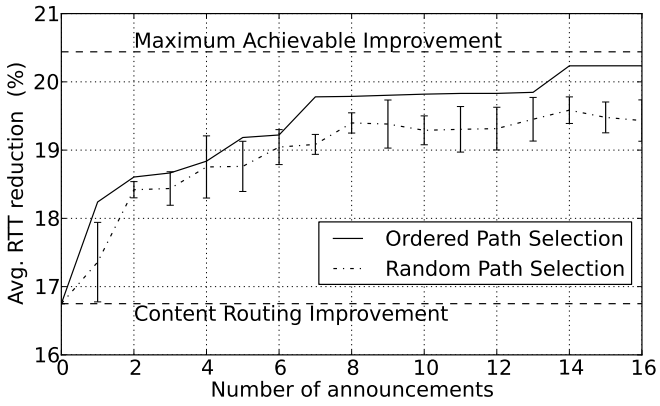


Figure 3.15: Average latency reduction with increasing number of route announcements per replica.

in this chapter, our experiments have evaluated the improvements that PECAN can provide by using 250 ingress routes at each geographic location. Over time, OSPs might be able to evaluate all of these 250 configurations, but doing so all at once is not practical. This warrants the question of how many routes an OSP needs to explore to achieve reasonable improvements from joint content and network routing.

When an OSP uses a limited set of routes to feed traffic to virtual replicas, it must decide which routes to use, but selecting the optimal subset of routes for each replica is computationally intractable. To avoid exploring all possible route combinations of route advertisements from each replica, we devise a simple heuristic: for each replica, we order the routes based on the average improvement they provide when compared to the performance over the default path. We then take the routes sequentially form that ordering and announce them from the replica. For example, if OSP decides to announce three of the alternate routes, it will pick three top routes from the ordered list. We compare this heuristic against selecting sets of routes at random.

Figure 3.15 shows how performance gains as we increase the number of announced routes. The gains in the figure are shown over the best replica baseline. The figure contains four plots: “maximum”, “ordered”, “random”, and “content”. The “Maximum” line represents the maximum gain that an OSP can get with joint routing. The “Ordered” line shows

the gains as we increase number of announced routes from zero to 16. The routes here are picked using the heuristic described above. The "Random" line shows the gains when we add additional routes at random. To generate the "random" plot, we run 10 iterations and report average and standard error values. Finally, the "content" line shows the content routing gains over the best replica. In most cases, the ordered heuristic outperforms the random route selection. It is also encouraging that only five additional virtual replicas per physical replica can obtain the approximately 60% of the gains one that are possible with the full set of routes.

3.5.4.2 Stability of replica selection

How many clients change their preferred replica after joint routing is added to content routed system? Content providers typically assign clients to replicas, taking into account loads at each replica. With PECAN, however, clients that previously had a suboptimal replica can shift their traffic to other replicas; this shifting might adversely affect loads on replicas, causing overloads. In fact, such imbalances could even increase latency.

If deploying PECAN on an existing content routing system creates such imbalances, its usage would not be practical: a provider would have to entirely reconfigure a replica setup to account for the change in traffic and load. The effect of joint routing on the stability of current replica choices is thus a genuine concern. Fortunately, we find that, when optimizing for latency, for 93% of the clients the choice of replica due to content routing does not change during joint routing. This suggests that PECAN is stable enough for practical deployment, since the loads on on each replica will not change significantly with a small number of moves.

3.6 Related Work

Content and network routing have been studied independently. In this section, we will review related work in improving content routing and network routing separately.

3.6.1 Content Routing

The late 1990s witnessed the first efforts in optimizing mapping of end-users to content or service replicas. Bhattacharjee *et al.* [26] presented a seminal paper describing a client-to-replica mapping system. This system used IP anycast to reach a directory service (*e.g.*, DNS) which then routed the clients to the best service replica based on the <client, replica> performance map. Seshan *et al.* [100] invented a new way to collect a comprehensive <client, replica> performance map: a small fraction of clients would be directed to randomly selected replicas to estimate the performance. Andrews *et al.* [29] presented a system called *Webmapper* that collected a <client, replica> performance map showcased algorithms for performing approximate client-to-best-replica matching.

The initial step in actual client to replica mapping is usually performed using Domain Name system (DNS). Pang *et al.* [89] evaluated the responsiveness of DNS to changes in client-to-replica mapping. Although the authors found that in many cases DNS is sluggish to respond, DNS, in most cases, is still the primary method for directing initial client requests to the best replicas. Huang *et al.* [60]introduced a DNS reflection method for client-to-replica performance map generation. Instead of usual actual client traffic, DNS reflection forces Local DNS (LDNS) servers to use iterative queries to remote replicas to estimate the delay between the LDNS servers and the replicas. The LDNS performance information is then used as a proxy metric for client-to-replica performance. Most recently, Wendell *et al.* [114] described a system called DONAR, which allows authoritative DNS servers to make client-to-replica mapping decisions with only partial global information.

3.6.2 Network Routing

We first discuss content routing techniques that require changes either in the end-systems or in significant parts of the Internet routers. Despite the benefits of such systems, they are still to see universal deployment. Then, we describe proposals that can operate within the constraints of today's networks and protocols.

Overlay Routing and Clean-Slate Network Routing Proposals. In the late 1990s, the overlay networks were a popular research topic. Informed Internet Routing and Transport Savage *et al.* [98] explored the benefits of an overlay routing system that selects best performing alternative paths. In addition to a conventional routing system underneath, the system requires an active network of overlay nodes to improve user experience. Andersen *et al.* [20] deployed and evaluated a similar overlay system across diverse locations in the Internet; commercial content distribution networks ultimately applied many of these techniques for finding the best paths to pre-cached content [78].

In mid 2000's, there were a number of proposals to improve routing in unconventional ways. Yang *et al.* [116] presented a system that can increase path diversity with routing deflections. End hosts in such systems can set bits that instructs routers on the path to perform deflections over better paths. In a similar spirit, Xu and Rexford [115] introduced MIRO: a system that provides increased diversity of paths choices for interdomain routing. Likewise, Motiwala *et al.* [80] presented a routing algorithm for Internet routers that enables scalable exploration of Internet path diversity. Unfortunately, utilizing such systems requires changes in Internet routers and in end-hosts.

Improvements to Conventional Internet Routing. In the last decade there has been a lot of research on wide-area routing, most of it focusing on the effectiveness of multi-homing enterprise networks. Our work builds upon these efforts and extends them to include scenarios where an OSP has a choice not only of diverse network paths (*network routing*) but also a choice of replicas (*content routing*). For example, Akella *et al.* [16, 17, 18, 19] explored the effects of multi-homing on the performance of a site that either sends or receives Internet traffic. The authors study 68 Akamai nodes in 17 cities as a testbed: Same city often contains multiple nodes, each with a different upstream ISP; the authors connect to all the Akamai nodes in one city to estimate performance of each ISP in that city, effectively

emulating a multi-homing setup in that city. The authors found that route optimization produces greater benefits in peak time intervals. Unfortunately, the study was limited in the number of alternative Internet paths (only upstreams) and it did not extend network routing to a logical conclusion of joint network and content routing.

Goldenberg *et al.* [53] assessed the benefit of single-site multihoming and also considered cost in the analysis. Guo *et al.* [57] analyzed a commercial solution for multihomed enterprises, which focused on possible performance gain with two upstream ISPs. Lee *et al.* [72] explored ways to scale active measurements for multihomed enterprises. Uhlig *et al.* [106] and Wang *et al.* [112] proposed formalizing upstream ISP selection as an optimization problem. Most of the efforts mentioned above focus on the enterprise setting and do not compare content routing with network routing.

3.7 Limitations and Future Work

In this section, we discuss some limitations of our current study and directions for future work. We focus in particular on how our dataset might be made more representative.

One of the most serious challenges in evaluating the performance gains than an OSPs can attain is to have a replica set that matches that of real OSPs. This equivalence entails two primary components: (1) diversity of replica locations and (2) diversity of route choice in each location. In terms of geographic diversity, our set of replicas is comparable to a set of North American Amazon EC2 data centers, although it is much smaller than the infrastructure that a large commercial OSP such as Google or Microsoft. In future work, we plan to perform similar experiments with replicas hosted across a tier-1 ISP backbone network; we are in the process of deploying this measurement infrastructure to allow for more comprehensive studies.

We were limited to exploring the routing diversity at each of our replicas that BGP

AS_PATH poisoning could provide. This method for exploring alternate routes could introduce unnecessary routing churn, but, in practice OSPs have a much wider variety of techniques at their disposal for controlling both ingress and egress routes. On the other hand, large OSPs might have many more ways to control their routing, both in egress and ingress directions. Most OSPs can choose among multiple egress routes to destinations and control ingress routing by selective announcements, BGP community attributes, or BGP AS_PATH prepending. The fact that real OSPs may have more route control mean that they may be able to realize even greater performance gains than we witnessed in our study of PECAN.

Another challenge in emulating real-world OSP performance is obtaining *a representative client set*. In our case the clients were PlanetLab nodes, which are hardly a representative set of the Internet users. Many of the nodes we used are housed in well connected university campuses. It does bias our client set, but it is not clear whether performance improvements— especially latency improvements—would differ with a more representative set of clients. On one hand, PlanetLab nodes might be more connected than average Internet nodes, this giving more route diversity to and from such nodes. On the other hand, one the routes PlanetLab nodes might be already well-provisioned and thus hard to improve on, while less connected and more remote networks might see more significant performance improvements from network routing.

A future study might also attempt to measure or approximate the overall user experience of using a particular replica and network route; user experience could possibly approximated by page load times, as has been done in previous work on Web performance [104]. Because our clients are run from PlanetLab nodes, it was not practical to instrument a browser and record the performance from each client. A promising direction for future work would be to conduct a more comprehensive study of how systems such as PECAN can improve user-perceived performance.

3.8 Summary

Online service providers today perform replica selection (content routing) and route selection (network routing) independently. We presented the design and evaluation of PECAN (Performance Enhancements with Content And Network routing), which performs joint content and network routing for a modern OSP. We design PECAN as an extension to content routing systems currently used by OSPs.

We evaluate the performance of PECAN on a globally distributed testbed emulating a modern OSP. We ran the replicated online service on five Transit Portal (TP) sites, each offering a large choice of network paths to our clients. We used 200 PlanetLab nodes as clients to estimate network performance to our replicated service. We find that PECAN achieves 22% more latency reduction than can be obtained by a modern content routing system alone, and that the gains remain significant as we increase the number of replicas. Finally, we find that PECAN can provide its benefit with only a few judiciously selected network paths: exploring just five sets of alternate network routes between clients and replicas in our testbed can yield 60% of the maximum possible benefit of joint routing to an online service provider.

CHAPTER IV

TIERED PRICING OF WIDE-AREA ROUTES

The increasing commoditization of Internet transit is changing the landscape of the Internet bandwidth market. Although residential Internet Service Providers (ISPs) and content providers are connecting directly to one another more often, they must still use major Internet transit providers to reach most destinations. These Internet transit customers can often select from among dozens of possible providers [90]. As major ISPs compete with one another, the price of Internet transit continues to plummet: on average, transit prices are falling by about 30% per year [84].

As a result of such competition, ISPs are evolving their business models and selling transit to their customers in many ways to try to retain profits. In particular, many transit ISPs implement pricing strategies where traffic is priced by volume or destination [55]. For example, most transit ISPs offer volume discounts for higher *commit levels* (*e.g.*, customer networks committing to a lower minimum bandwidth receive a higher per-bit price quote than customers committing to a higher minimum bandwidth [84]). Such a market is said to implement *tiered pricing* [67]. Through private communication with network operators, we identified many other instances of tiered pricing already being implemented by ISPs. These pricing instruments involve charging prices on traffic bundles based on various factors, such as how far the traffic is traveling, and whether the traffic is "on net" (*i.e.*, to that ISP's customers) or "off net". Still, we understand very little about the extent to which tiered pricing benefits both ISPs and their customers, or if there might be better ways to structure the tiers. In this chapter, we study *destination-based tiered pricing*, with the goal of understanding how ISPs should bundle and price connectivity to maximize their profit.

In this study, we grapple with the balance between the prescriptions of economic theory

and a variety of practical constraints and realities. On one hand, economic theory says that higher market granularity leads to increased efficiency [74]. Intuitively, an Internet transit market that prices individual flows is more efficient than one that sells transit in bulk, since customers pay only for the traffic they send, and since ISPs can price those flows according to their cost. On the other hand, various practical constraints prevent Internet transit from being sold in arbitrarily fine granularities. Technical hurdles and additional overhead can make it difficult to implement tiered pricing in current routing protocols and equipment. Additionally, tiered pricing can be more difficult for wholesale customers to understand if there are too many tiers. Transit ISPs would ideally like to come close to maximizing their profit with only a few pricing tiers, since implementing more pricing tiers introduces additional overhead and complexity. Our analysis shows that, indeed, in many cases, an ISP reaps most of the profit possible with infinitesimally fine-grained tiers using only two or three tiers, assuming that those two or three tiers are structured properly.

Although understanding the benefits of different pricing structures is important, modeling them is quite difficult. The model must take as an input existing customer demand and predict how traffic (and, hence, ISP profit) would change in response to pricing strategies. Such a model must capture how customers would respond to any pricing change—for any particular traffic flow—as well as the change in cost of forwarding traffic on various paths in an ISP's network. Of course, many of these input values are difficult to come by even for network operators, but they are especially elusive for researchers; additionally, even if certain values such as costs are known, they change quickly and differ widely across ISPs.

The model we develop allows us to estimate the relative effects of pricing and bundling scenarios, despite the lack of availability of precise values for many of these parameters. The general approach, which we describe in Section 4.2, is to start with a demand and cost model and assume both that ISPs are already profit-maximizing and that the current prices reflects both customer demand and the underlying network costs. These assumptions allow us to either fix or solve for many of the unknown parameters and run counterfactuals to

evaluate the relative effects of dividing the customer demand into pricing tiers. To drive this model, we use traffic data from three real-world networks: a major international content distribution network with its own network infrastructure; an European transit ISP; and an academic research network. We map the demand and topology data from these networks to a model that reflects the service offerings that real-world ISPs use.

Using our model, we evaluate three scenarios:

- *What happens if an ISP increases the number of tiers for which it sells transit?* We find that profit increases, but the returns diminish as the number of tiers increases: with 3–4 tiers, it is possible to capture 90–95% of the profit that could be captured with an infinite number of granularities, assuming that these tiers are divided in the right way.
- *How do strategies for dividing capacity into distinct bundles and pricing those bundles affect an ISP's profit?* Our analysis shows that ISPs must judiciously choose how they divide traffic into pricing tiers. A naïve approach (*e.g.*, based only on traffic cost or on demand) might require dozens of pricing tiers to capture most of the possible profit. We find that dividing traffic into tiers in a way that accounts for *both traffic demand and the cost of carrying traffic* yields more profit than the current practice that is based only on cost, and is nearly as effective as an optimal division.
- *How do the benefits of various pricing strategies depend on the network topology and traffic demands?* We find that networks with high variability in *cost* of delivering traffic obtain greater benefit from bundling. We also observe that networks with high variability in *demand* require more bundles to capture maximum profit.

We evaluate these and other questions across two customer demand models, four network cost models, and a range of input parameters, such as price sensitivity. Although each of the models might not be perfectly accurate, they yield results that are consistent both across models and with our intuition about markets.

We make three contributions. First, we taxonomize the state of the art and trends in

pricing instruments for Internet transit (Section 4.1). Second, to analyze the effects of tiered pricing, we develop a model that captures demands and costs in the transit market. One of the challenges in developing such a model is applying it to real traffic data, given many unknown parameters (*e.g.*, the cost of various resources, or how users respond to price). Hence, we develop methods for fitting empirical traffic demands to theoretical cost and demand models. We use this approach to evaluate ISP profit for pricing strategies under a range of possible cost models and network topologies (Section 4.2). Third, we apply our model to real-world traffic matrices and network topologies to characterize a range of simple bundling strategies that are close to optimal (Section 4.3); we also suggest how these strategies could be implemented in practice (Section 4.4).

4.1 Background

In this section, we describe the current state of affairs in the Internet transit market. We first taxonomize *what* services (bundles) ISPs are selling. We then provide intuition on *why* ISPs are moving towards tiered wholesale Internet transit service.

4.1.1 Current Transit Market Offerings

Unfortunately, there is not much public information about the wholesale Internet transit market. ISPs are reluctant to reveal specifics about their business models and pricing strategies to their competitors. Therefore, to obtain most of the information in this section, we engaged in many discussions and email exchanges with network operators. Below, we classify the types of Internet transit service we identified during these conversations. Although much of the information in this section is widely known in the network operations community, it is difficult to find a concise taxonomy of product offerings in the wholesale transit market. The taxonomy below serves as a point of reference for our discussions of tiered pricing in this chapter, but it may also be useful for anyone who wishes to better understand the state of the art in pricing strategies in the wholesale transit market.

Peering business relationships (or, more formally, *settlement-free peering* relationships) have been extensively studied by networking researchers [32, 41, 46, 48, 66] and well-documented in the industry white papers [84]. Most peering connections are established through public Internet eXchange Points (IXP), while higher bandwidth peering often requires private peering sessions. A network engaging in a settlement free peering allows its peer to reach *on-net* destinations—destinations in its own network, and destinations in customer networks. For the peering to be settlement-free, most ISPs pose a set of requirements to prospective peers, such as number of interconnection points, geographic coverage, or ingress/egress traffic ratios. If an ISP cannot meet peering requirements, it is forced to buy Internet *transit* or *paid peering* [28, 71, 92].

Transit. Most ISPs offer conventional Internet transit service. Internet transit is sold at a *blended rate*—a single price (usually expressed in $/Mbps/month)—charged for traffic to all destinations. Historically, blended rates have been decreasing by 30% each year [84]. Blended rate is the simplest and yet the most crude way to charge for traffic. If network costs are highly variable, less costly flows in the blended-rate bundle subsidize other, more expensive flows. ISPs often innovate by offering more than one rate: We summarize three pricing models that require two or more rates: (1) *paid peering*, (2) *backplane peering*, and (3) *regional pricing*.

Paid peering is similar to settlement-free peering, except that one network pays to reach the other. A major ISP might separately sell *off-net* routes (wholesale transit) at one rate and *on-net* routes (to reach destinations inside its own network) at another (usually lower) rate. For example, national ISPs in Eastern Europe, Australia, and in other regions may sell local connectivity at a discount to increase demand for local traffic, which is is significantly cheaper than transit to outside global destinations [13]. The on-net routes are also offered at a discount by some major transit ISPs to large content providers, because such transit ISPs can recoup part of the costs from their customers, who congest paid upstream links

to transit ISPs by downloading the content. Some instances of paid peering have spawned significant controversy: most recently, Comcast—primarily a network serving end-users—was accused of a network neutrality violation when it forced one tier-1 provider to pay to reach Comcast's customers [73].

Backplane peering occurs when an ISP, in addition to selling global transit through its own backbone, charges a discount rate for the traffic it can offload to its peers at the same Internet exchange. Smaller ISPs buy such a service because they might not meet all the settlement-free peering requirements to peer directly with the ISPs in the exchange. Although many large ISPs discourage this practice, some ISPs deviate by offering backplane peering to retain customers or to maintain traffic ratios with their peers. As with paid peering, the ISP selling backplane peering has to account and charge for at least two traffic flows: one to peers and another to its backbone.

Regional pricing occurs when transit service providers offer different rates to reach different geographic regions. The regions can be defined at different levels of granularity, such as PoP, metro area, regional area, nation, or continent. In some instances, the transit ISP offers access to all regions with different prices; in other instances, the downstream network purchases access only to a specific geographic region (*e.g.*, access only to South America or Australia). In practice, due to the overhead of provisioning and maintaining many sessions to the same customer, ISPs rarely use more than one or two extra price levels for different regions.

We speculate that the bundling strategies described above arose primarily from operational and cost considerations. For example, it is relatively easy for a transit ISP to tag which routes are coming from customers and which routes are coming from peers and then in turn sell them separately to its customers. Similarly, it is relatively easy to sell local (*i.e.*, less costly) routes separately. We show that these naïve bundling strategies might not be as effective as bundling strategies that account for both cost and demand.

4.1.2 The Trend Towards Tiered Pricing

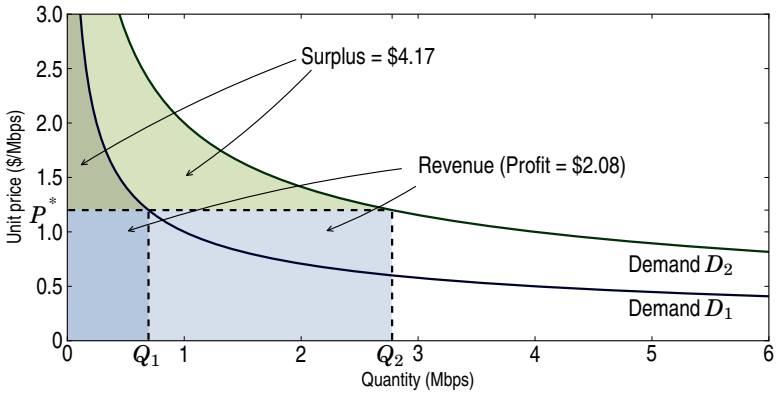
Conventional blended rate pricing is simple to implement, but it may be inefficient. ISPs can lose profit as a result of blended-rate pricing, and customers can lose surplus. This is an example of *market failure*, where goods are not being efficiently allocated between participants of the market. Another outcome of blended-rate pricing is the *increase in direct peering* to circumvent "one-size-fits-all" transit. Both phenomena provide incentives for ISPs to improve their business models to retain revenue. We now explain each of these outcomes.

4.1.2.1 Profit and surplus loss

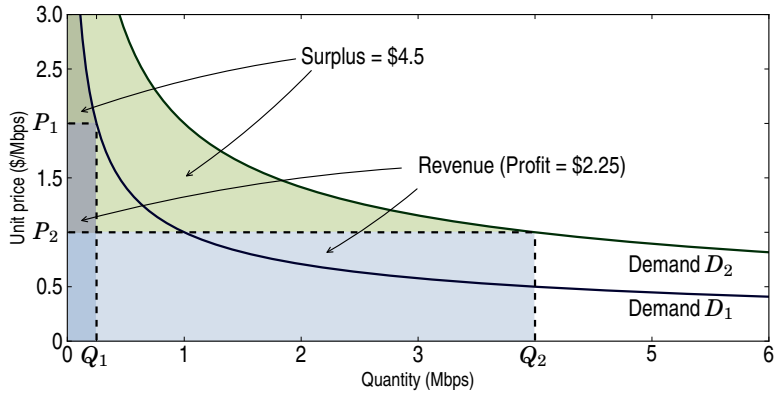
Selling transit at a blended rate could reduce profit for transit ISPs and surplus for customers. We define an ISP's *profit* as its revenue minus its costs, and *customer surplus* as customer utility minus the amount it pays to the ISP. Unrealized profit and surplus can occur when ISPs charge a single price per bandwidth unit while incurring different costs when delivering traffic to destinations.

Figure 4.1 illustrates how tiered pricing can increase both the profit for an ISP and the surplus for a customer. The downward-sloping curves represent consumer demand[1] to two destinations. Since the demand slope D_2 is higher than demand slope D_1, the customer has higher demand for the second destination in the ISP's network. Assume that the ISP cost of serving demand D_1 is \$1, while the cost of serving demand D_2 is \$0.5. Modeling demand with constant elasticity (Section 4.2), the *profit maximizing price* can be shown to be $P_0 = \$1.2/Mbps$. If, however, the ISP is able to offer two bundles, then the profit maximizing prices for such bundles would be $P_1 = \$2.7$ and $P_2 = \$1$. Figure 1(b) shows that this price setup not only increases ISP profit but also increases consumer surplus and thus social welfare.

[1]We model consumer demand as *residual demand*. Residual demand accounts for consumption change both due to inherent consumer demand and due to some consumers shifting consumption to substitutes, such as other ISPs (See Section 4.2.2.1.)



(a) **Blended-rate pricing.** ISP charges a single blended rate P_0.



(b) **Tiered pricing.** ISP charges rates P_1 and P_2 for flows.

Figure 4.1: Market efficiency loss due to coarse bundling.

The market achieves higher efficiency because customers adjust their consumption levels of the ISP network according to their demand and to the prices that the ISP exposes, which directly depend on its costs. Without the ISP's indirect exposure of its costs, the customer consumes less of the cheaper capacity and more of the expensive capacity than it would otherwise. In Section 4.2, we formalize the market that we have used in this example and propose more complex demand and cost models.

4.1.2.2 Increase in direct peering

Charging for traffic at a blended rate also provides incentives for client networks to connect directly to georgraphically close Internet Exchange Points (IXPs). For instance, if a transit ISP charges only blended rate, client network might find geographically close IXPs cheaper

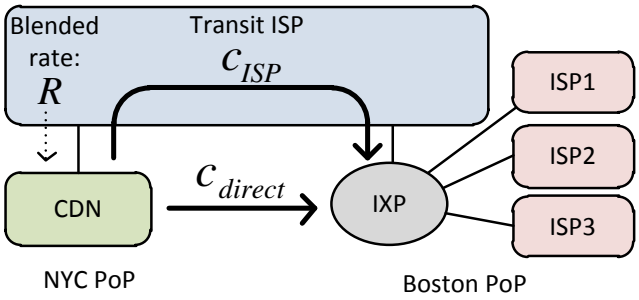


Figure 4.2: The customer procures a direct link if the cost for such a link is lower than the blended rate $c_{direct} < R$.

to reach by leasing or purchasing private links. While direct peering is generally perceived as a positive phenomenon, for transit ISPs it means less revenue. Direct peering efforts can also diminish economies of scale: instead of using shared ISP infrastructure, customers provision their own connectivity to IXPs.

Figure 4.2 illustrates an interaction between an upstream ISP and a CDN client (*e.g.*, Google, Microsoft) with its own backbone, which extends to the NYC PoP. The CDN might, or might not, have a content cache at the Boston IXP, but since it does not have its own backbone presence at the IXP, the CDN must pay the upstream ISP to reach it. The ISP offers a blended rate R at the the NYC PoP for all the traffic, including the traffic to the Boston IXP. The blended rate R is set to compensate the upstream provider for the overall traffic mix and, therefore, is *higher* than the amortized cost of most of the cheaper (more localized) flows that ISP is serving (*i.e.*, the flows between the NYC and Boston PoPs). The CDN eventually will procure a direct link to the Boston IXP, if it finds that it can procure such a direct link at an amortized cost $c_{direct} < R$. Assuming the ISP's profit margin is M and flow accounting overhead is A (discussed in Section 4.4.2), such a direct link presents a *market failure* if $c_{direct} > (M+1)c_{ISP} + A$, because the customer deploys additional capacity at a higher cost than the ISP could have charged in a tiered market.

Some operators we interviewed confirm that they periodically re-evaluate transit bills and expand their backbone coverage if they find that having own presence in an IXP pays off. In today's transit market, many customers increasingly opt for direct peering [69];

transit service providers are absorbing losses as a result of competitive pressure [84]. Naturally, this pressure increases the incentive for ISPs to adopt a tiered pricing model for local traffic. The central question, then, is how they should go about structuring these tiers. The rest of the chapter focuses on this question.

4.2 Modeling Profits, Costs, and Demands

We develop demand and cost models that capture ISP profit under various pricing strategies. Although we doubt there is a perfect model for demand in the Internet transit market, we perform our evaluation with two common demand models. Because cost is also difficult to model, we devise four network cost models. We first define ISP profit and then describe demand and cost models.

4.2.1 ISP Profit

We consider a transit market with multiple ISPs and customers. Each ISP is rational and maximizes its profit, which we express as the difference between its revenue and costs:

$$\Pi(\vec{P}) = \sum_{p_i \in \vec{P}} \left(p_i Q_i(\vec{P}) - c_i Q_i(\vec{P}) \right) \tag{4.1}$$

where p_i is the price an ISP sets to deliver flow i, c_i is the unit cost for i, and $Q_i(\vec{P})$ is the demand for i given a vector of prices $\vec{P} = (p_1, p_2, \ldots, p_n)$. An ISP chooses the price vector $\vec{P}$ that maximizes its profit. Having a price for each flow allows us to explore different pricing strategies by bundling flows in different ways. For example, blended rate pricing requires p_i to be equal for all i; we can explore different tiered pricing approaches by requiring various subsets of all flows to have the same price.

Given knowledge of both the traffic demand of customers and the costs associated with delivering each flow, we can compute ISP profit. Unfortunately, it is difficult to validate any particular demand function or cost model; even if validation were possible, it is likely that cost structures and customer demand could change or evolve over time. Accordingly,

we evaluate ISP profit for various tiered pricing approaches under a variety of demand functions and cost models. Section 4.2.2 describes the demand functions that we explore, and Section 4.2.3 describes the cost models that we consider.

4.2.2 Customer Demand

To compute ISP profit for each pricing scenario, we must understand how customers adjust their traffic demand in response to price changes. We consider two families of demand functions: *constant elasticity* and *logit*.

Constant elasticity demand. The constant elasticity demand (CED) is derived from the well-known *alpha-fair* utility model [79], which is often used to model user utility on the Internet. The alpha-fair utility takes the form of a concave increasing utility function, which emulates a decreasing marginal benefit to additional bandwidth for a user. In this model flow demands are *separable* (*i.e.*, changes in demand or prices for one flow have no effect on demand and prices of other flows). The CED model is most appropriate for scenarios when consumers have no alternatives (*e.g.*, when the content that a customer is trying to reach is not replicated, or the customer needs to communicate with a specific endpoint on the network).

Logit demand. To capture the fact that customers might sometimes have a choice between flows (*e.g.*, sending traffic to alternative destination if the current one becomes too expensive), we also perform our analysis using the *logit model*, where demands are not separable: the price and demand for any flow depend on prices and demands for the other flows. The logit model is frequently used for this purpose in econometric demand estimation [76]. In the logit model, each consumer nominally prefers the flows that offers the highest utility. This matches well with scenarios when consumers have several alternatives (*e.g.*, when requested content is replicated in multiple places).

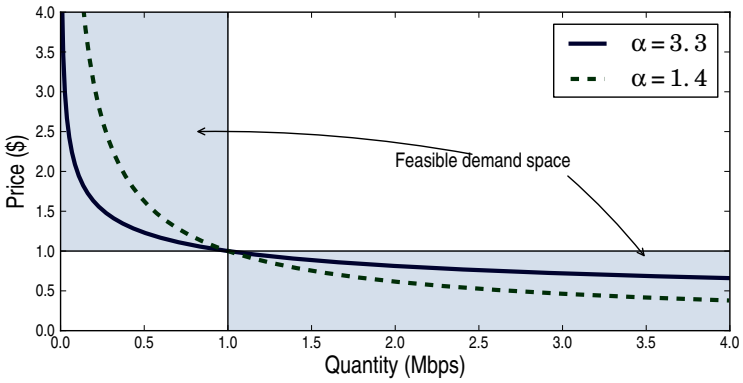


Figure 4.3: Feasible CED demand functions for $v = 1$.

4.2.2.1 Constant elasticity demand

The CED demand function is as follows:

$$Q_i(p_i) = \left(\frac{v_i}{p_i}\right)^{\alpha} \tag{4.2}$$

where p_i is the unit price (*e.g.*, \$/Mbit/s), $\alpha \in (1, \infty)$ is the price sensitivity, and $v_i > 0$ is the valuation coefficient of flow i. The demand function can be interpreted to represent either *inherent* consumer demand or *residual* consumer demand, which reflects not only the inherent demand but also the availability of substitutes.

Figure 4.3 presents example CED demand functions for $v = 1$ and two values of α, 3.3 and 1.4. Higher values of α indicate high elasticity (users reduce use even due to small changes in price). For example, the demand with elasticity $\alpha = 3.3$ might represent the traffic from residential ISPs, who are more sensitive to wholesale Internet prices and who respond to price changes in a more dramatic way. Similarly, the demand with elasticity $\alpha = 1.4$ might represent the traffic from enterprise customers, who are less sensitive to the Internet transit price changes. Although our model does not capture full dynamic interaction between competing ISPs (*e.g.*, price wars), modeling demand as residual allows us to account for the existing competitive environment and switching costs. As discussed above, high elasticity can also indicate that competitors are offering more affordable substitutes, and that switching costs for customers are low. In our evaluation, we use a range of price

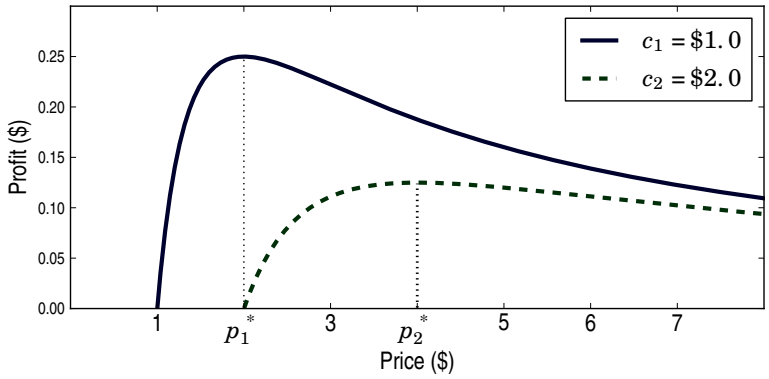


Figure 4.4: Profit for two flows with identical demand ($v_1 = v_2 = 1.0$, $\alpha = 2$) but different cost.

sensitivity values to measure how ISP profit changes for different values of the elasticity of user demand. The gray area in Figure 4.3 shows that we can cover all feasible demand functions simply by varying the sensitivity parameter.

CED profit. Using the expressions for ISP profit (Equation 4.1) and demand (Equation 4.2), and assuming separability of demand of different flows, the ISP profit is:

$$\Pi(\vec{P}) = \sum_{p_i \in \vec{P}} \left(\frac{v_i}{p_i} \right)^{\alpha} (p_i - c_i) \,. \tag{4.3}$$

CED profit-maximizing price. By differentiating the profit, we find the profit-maximizing price for each flow i:

$$p_i^* = \frac{\alpha c_i}{\alpha - 1}. \tag{4.4}$$

CED consumer surplus. Consumer surplus is the difference between consumer utility and the price paid. Price at the equilibrium is equal to the marginal utility, and thus we can find utility by integrating price as a function of demand ($p_i = v_i / q_i^{1/\alpha}$, Equation 4.2) in terms of q_i. Substituting in the resulting equation quantity with price and substracting the price paid, we get consumer surplus expression as function of price:

$$CS(\vec{P}) = \sum_{i=1}^{n} \left(\frac{\alpha v_i^{\alpha} p_i^{1-\alpha}}{\alpha - 1} - p_i \right) \tag{4.5}$$

Figure 4.4 illustrates profit maximization for two flows that have identical demand functions but different costs. For example, the first flow costs $c_1 = \$1.0$ per unit to deliver and mandates optimal price $p^* = \$2.0$ which results in \$0.25 profit. The second flow is more costly thus the profit maximizing price is higher. In this case, the first plot might represent profit for local traffic, while the second plot represents national traffic: ISPs must price national traffic higher than local-area traffic to maximize profit.

CED price for bundled flows. In our evaluation, we test various pricing strategies that bundle multiple flows under the same profit-maximizing price. To find the price for each bundle, we first map real world demands to our model to obtain the valuation v_i and cost c_i for each flow. Then, we differentiate the profit (Equation 4.3) with respect to the price of each bundle. For example, when we have a single bundle for all flows, we obtain the following profit-maximizing price:

$$P^* = \frac{\alpha \sum_{i=1}^{n} c_i v_i^{\alpha}}{(\alpha - 1) \sum_{i=1}^{n} v_i^{\alpha}} \tag{4.6}$$

where n is the number of flows. Section 4.3 details this approach.

4.2.2.2 Logit demand

The logit demand model assumes that each consumer faces a discrete choice among a set of available goods or services. In the context of data transit, the choice is between different destinations or flows. Following Besanko *et al.* [24], a consumer j using flow i will obtain the net utility (surplus):

$$u_{ij} = \alpha(v_i - p_i) + \varepsilon_{ij}$$

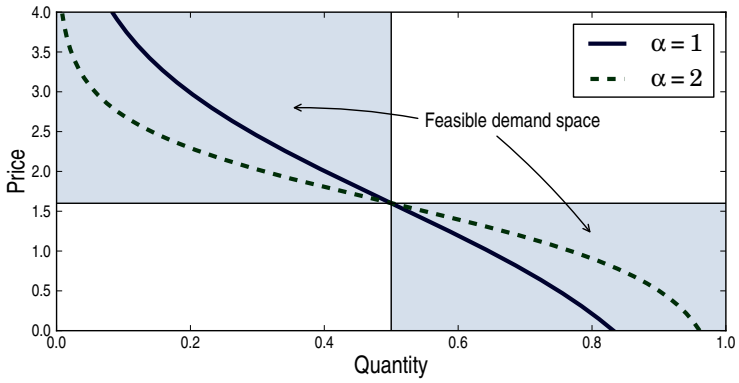


Figure 4.5: Logit demand function.

where $\alpha \in (0, \infty)$ is the elasticity parameter, v_i is the "average" consumer's *maximum willingness to pay* for flow i, p_i is a price of using i, and ε_{ij} represents consumer j's idiosyncratic preference for i (where ε_{ij} has a Gumbel distribution.) The logit model defines the probability that any given consumer will use flow i as a function of the price vector of all flows:

$$s_i(\vec{P}) = \frac{e^{\alpha(v_i - p_i)}}{\sum_j e^{\alpha(v_j - p_j)} + 1} \tag{4.7}$$

where $\sum_i s_i(\vec{P}) = 1$. The demand for flow i equals the product of $s_i(\vec{P})$ and the total number of consumers (K):

$$Q_i(\vec{P}) = K s_i(\vec{P}). \tag{4.8}$$

Here, s_i is also called the *market share* of flow i. The model also accounts for the possibility that some customers elect not to send traffic to any destination. The market share for traffic not sent is:

$$s_0(\vec{P}) = \frac{1}{\sum_j e^{\alpha(v_j - p_j)} + 1}.$$

Figure 4.5 shows examples of logit demand functions. We assume a setting with two flows, with two values for the valuation v_i, 1.6 and 1. We fix the price for the first flow to

1, and we vary the price for the second flow between 0 and 4. The figure shows demand curves for the second flow, for two values of α. Similar to the constant elasticity demand model, lower values of α indicate low elasticity of demand, where users need bigger price variations to modify their usage.

Logit profit. Using the expressions for ISP profit (Equation 4.1) and logit demand (Equation 4.8), the ISP profit is:

$$\Pi(\vec{P}) = K \sum_{p_i \in \vec{P}} s_i(\vec{P})(p_i - c_i). \tag{4.9}$$

Logit profit-maximizing prices. To find the profit maximizing price for flow i, we find the first-order conditions for Equation 4.9:

$$p_i^* = c_i + \frac{1}{\alpha s_0}. \tag{4.10}$$

Due to the presence of s_0, p_i^* recursively depends on itself and on profit-maximizing prices of other flows. To obtain maximum profit, we develop an iterative heuristic based on gradient descent that starts from a fixed set of prices ($p_i = P_0, \forall i$) and greedily updates them towards the optimum.

Logit consumer surplus. After ISP sets profit-maximizing prices $\vec{P}$, we can compute consumer surplus. We find consumer surplus expression by taking the expectation of the sum of all the consumer utilities:

$$CS(\vec{P}) = K \frac{\gamma + \ln(\sum_{i=1}^{n} e^{\alpha(v_i - p_i)} + 1)}{\alpha} \tag{4.11}$$

Valuation and cost of bundled flows. To test pricing strategies, we first map real traffic demands to the model to find the valuation v_i and cost c_i for each flow i. We then bundle

the flows as described in Section 4.3.2.1. Knowing that $\sum_i s_i = 1$ and applying Equation 4.7 allows us to compute valuations for any bundle of flows as:

$$v_{bundle} = \frac{\ln\left(\sum_{i=1}^{n} e^{\alpha v_i}\right)}{\alpha} \tag{4.12}$$

where v_i are valuations of the flows in the bundle. Similarly we can find the average unit cost of combined flows in each bundle:

$$c_{bundle} = \frac{\sum_{i=1}^{n} c_i e^{\alpha v_i}}{\sum_{i=1}^{n} e^{\alpha v_i}}. \tag{4.13}$$

4.2.3 ISP Cost

Modeling cost is difficult: ISPs typically do not publish the details of operational costs; even if they did, many of these figures change rapidly and are specific to the ISP, the region, and other factors. To account for these uncertainties, we evaluate our results in the context of several cost models. We also make the following assumptions. First, we assume the more traffic the ISP carries, the higher cost it incurs. Although on a small scale the bandwidth cost is a step function (the capacity is added at discrete increments), on a larger scale we model cost as a linear function of bandwidth. Second, we assume that ISP transit cost changes with distance. Both assumptions are motivated by practice: looking only at specific instances of connectivity, the cost is a step function of distance (*e.g.*, equipment manufacturers sell several classes of optical transceivers, where each more powerful transceiver able to reach longer distances costs progressively more than less powerful transceivers [36]). Over a large set of links, however, we can model cost as a smooth function of distance.

The cost models below offer only *relative* flow-cost valuations (*e.g.*, flow A is twice as costly as flow B); they do not operate on absolute costs. These relative costs must be reconciled with the blended prices used to derive customer valuations. We describe methods for reconciling these values in Section 4.3.1. Each cost model has a generic tuning

parameter, denoted as θ, which we use in the evaluation.

Linear function of distance. The most straightforward way to model ISP's costs as a function of distance is to assume cost increases linearly with distance. Although, in some cases, this model does not hold (*e.g.*, crossing a mountain range is more expensive than crossing a region with flat terrain), we often observe that ISPs charge linearly in the distance of communication [27, 35]. As we model cost as linear function of distance, we set cost $c_i = \gamma d_i + \beta$, where γ is a scaling coefficient that translates from relative to real costs, β is a base cost (*i.e.*, the fixed cost that the ISP incurs for communicating over any distance), and d_i is the geographical distance between the source and destination served by an ISP. We describe how we determine γ in Section 4.3.1. We model the base cost β as a fraction of the maximum cost without the base component. More formally: $\beta = \theta \max_{j \in 1 \cdots n} \gamma d_j$, where θ in this cost model is a relative base cost fraction, and n is number of flows with different cost. For example, given distances 1, 10, and 100 miles, $\gamma = \$1/mile$, and $\theta = 0.1$, the resulting base cost β is \$10, and thus flow costs are \$11, \$20, and \$110. In the evaluation, we vary θ to observe the effects of different base costs. For example, low θ values (low base cost) here represent a case where link distance is the largest contributor to the total service cost.

Concave function of distance. We are also aware of ISPs that price transit as a concave function of distance [63, 85]. For this scenario we model the ISP's cost as $c_i = \gamma(a \log_b d_i + c) + \beta$. Figure 4.6 shows a concave curve fitting to two price data sets, resulting in $a \approx 0.5$, $b \approx 6$, and $c \approx 1$ for normalized prices and distance. As in the case of linear cost, we set the base offset cost $\beta = \theta \max_{j \in 1 \cdots n} \gamma(a \log_b d_i + c)$. We use θ in the evaluation to change link distance contribution to the total cost.

Function of destination region. As described in Section 4.1.2, both private communication with network operators and publicly available data suggest that ISPs can also charge

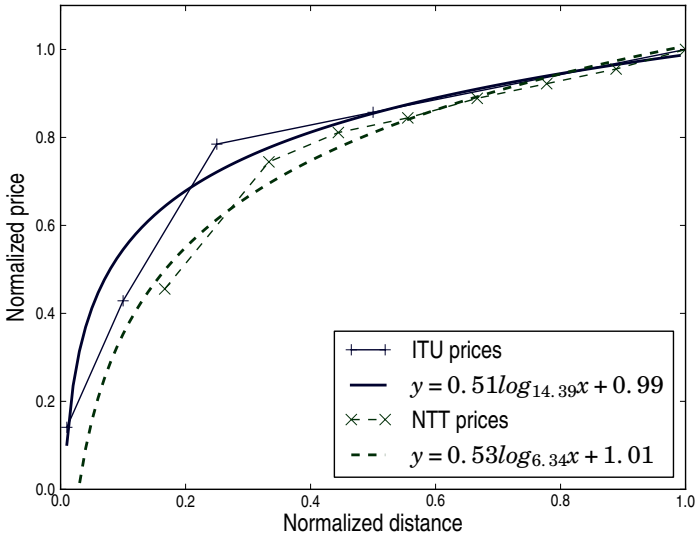


Figure 4.6: Using price data from ITU [63] and NTT [85] to fit concave distance to cost mapping curve.

for traffic at rates depending on the region where traffic is destined [44, 88, 105]. For example, an ISP might have less expensive capacity in the metropolitan area than in the region, less expensive capacity in the region than in the nation, and less expensive capacity in the nation than across continental boundaries. We divide flows into three categories: *metropolitan*, *national*, and *international*. We map the flows into these categories by using data from the GeoIP [75] database: flows that originate and terminate in the same city are classified as metro, and flows that start and end in the same country are classified as national; all other flows are classified as international. For EU ISP we only have distances between traffic entry and exit points, thus we classify flows traveling less than 10 miles as metro, flows that travel less than 100 miles as national, and longer flows as international. We set the costs as follows: $c_{metro} = \gamma$, $c_{nation} = \gamma 2^{\theta}$, and $c_{int} = \gamma 3^{\theta}$. This form allows us to test scenarios when there is no cost difference between regions ($\theta = 0$), the cost differences are linear ($\theta = 1$), and costs are different by magnitudes ($\theta > 1$).

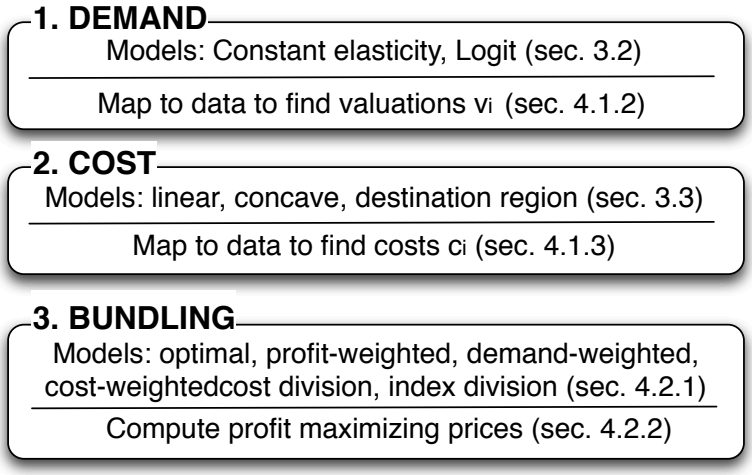


Figure 4.7: We evaluate the effect of tiered pricing on Internet transit by separately modeling the demand and cost of traffic and the way ISPs bundle flows under the same price. At each step we use real-world data to derive unknown parameters.

Function of destination type. As we described in Section 4.1.2, ISPs offer discounts for the traffic destined to their customers ("on net" traffic), while charging higher rates for traffic destined for their peers ("off net" traffic). These offerings are motivated by the fact that ISPs do recover some of their transport cost for the traffic sent to other customers. In our evaluation, we model this cost difference by setting the cost of the traffic to peers to be twice as costly than traffic to other customers. The logic behind such a model is that when an ISP sends the traffic between two customers, it gets paid twice by both customers, but when an ISP sends traffic between a customer and a peer it is only paid by the customer. The parameter θ indicates a fraction of traffic at each distance that is destined to clients, as opposed to traffic that is destined to peers and providers.

4.3 Evaluating Tiered Pricing Strategies

In this section, we evaluate the efficiency of destination-based tiered pricing using the model presented in Section 4.2 and real topology and demand data from large networks. Our goal is to understand how the consumer suprlus and the profit that an ISP extracts from

offering tiered-pricing depends on the number of tiers (or bundles), the bundling strategy used, and the network topology and traffic demand.

One of the major challenges we face is that we cannot know some aspects of the cost and demand models, or even which model to use. We use the ISP data to derive model parameters, such as valuation or cost, and evaluate the profit of each strategy across models and input parameters. Figure 4.7 presents an overview of our approach for computing ISP profit and consumer surplus.

Our evaluation yields several important results. First, we show that an ISP needs only 3–4 bundles to capture 90–95% of the profit provided by an infinite number of bundles, if it bundles the traffic appropriately. Second, choosing a bundling strategy that considers both flow demand and cost is almost as effective as an exhaustive search for the best combination of bundles. Finally, we observe that the topology and traffic of a network influences its bundling strategies: networks with higher coefficient of variation of demand need more bundles to extract maximum profit.

4.3.1 Mapping Data to Models

Because we do not know the parameters that we need to compute the profit-maximizing prices and the maximum profit for each bundling strategy, we must derive them. We first describe the data and how we extract the necessary information for computing model parameters. Then, we show how to apply the demand models to the real traffic demands to compute the valuation coefficients v_i for each flow i. Finally, we derive the ISP's cost for servicing the flow by applying flow distance information to each of the cost models.

4.3.1.1 Data Sources

We use demand and topology data from three networks: a European ISP serving thousands of business customers (EU ISP), one of the largest CDN providers in the world (CDN), and a major research network in United States (Internet 2). The data consists of sampled NetFlow records from core routers in each network for 24 hours. Table 4.1 presents more

Table 4.1: Data sets used in our evaluation. The columns represent: the network, data capture date, demand-weighted average of flow distances, coefficient of variation (CV) of flow distances, aggregate traffic per second, and CV of demand of different flows.

Data set	Date	Distance (miles)		Traffic (Gbps)	
		w-avg	CV	Aggregate	CV
EU ISP	11/12/09	54	0.70	37	1.71
CDN	12/02/09	1988	0.59	96	2.28
Internet 2	12/02/09	660	0.54	4	4.53

details about the data sets.

To drive our model, we must compute the traffic volume (which captures consumer demand) and the distance between the source and destination of each flow (which captures the relative cost of transit). To do so, we extract the source and destination IP and port information, as well as the traffic level, from each NetFlow record. We obtain the demand for each flow by aggregating all records of the flow, while ensuring that we do not double-count records that are duplicated on different routers.

To compute distances that reflect the ISP's cost of sending traffic, we use the following heuristics. For the EU ISP, the distance that each flow travels in the ISP's network is the geographical distance between the flow's entry and exit points, whose identity and location is known. For the CDN, we use the GeoIP database [75] to estimate the distance to the destination. Although this may not reflect the real distance that a packet travels (because part of the path may be covered by another ISP), we assume that it is still reflective of the cost incurred by the CDN. Finally, for Internet2, because each flow may traverse multiple routers, we use the port information to identify the links the flow has traversed. The distance each flow traverses is the sum of the links in the path, where the link length is the geographical distance between the neighboring routers.

4.3.1.2 Discovering valuation coefficients

The valuation coefficient v_i indicates the valuation of flow i. We need the valuation coefficeint v_i to capture how the demand for flow i varies with price (Equations 4.2 and 4.8) and thus affect the ISP profit. To find v_i for each flow, we assume that ISPs charge the same

blended price P_0 for each flow and map the observed traffic demand from the data to each demand model.

CED valuation coefficient. From Equation 4.2 we obtain:

$$v_i = \frac{q_i^{\left(\frac{1}{\alpha}\right)}}{P_0}$$

where q_i is observed demand on flow i and α is the sensitivity coefficient that we vary in the evaluation.

Logit demand valuation coefficient. Starting with Equation 4.8 and knowing that $s_0 + \sum s_i = 1$ and $s_0 = \frac{1}{\sum e^{\alpha(v_j - P_0)} + 1}$, we obtain:

$$v_i = \frac{\log s_i - \log s_0}{\alpha} + P_0$$

where s_i is the market share of flow i. We vary s_0 in the evaluation and compute the remaining market shares from observed traffic as $s_i = \frac{q_i(1-s_0)}{\sum q_i}$.

4.3.1.3 Discovering costs

To estimate the ISP profit, we must know the cost that an ISP incurs to service each flow. However, the data provides information only about the distance each flow traverses, which reflects only the relative cost (*e.g.*, flow A is twice as costly as flow B), rather than an absolute cost value. To normalize the cost of carrying traffic to the same units as the price for the flow, we introduce a scaling parameter γ, where $c_i = \gamma f(d_i)$ and d_i is the distance covered by i (see Section 4.2.3). For each demand model, we compute the scaling parameter by assuming (as in computing valuation coefficients) that ISPs are rational and profit maximizing, and charge the same price P_0 for each flow.

CED. Using Equation 4.6 and substituting c_i for $\gamma f(d_i)$, we find:

$$\gamma = \frac{P_0(\alpha - 1)\sum_{i=1}^{n} v_i^{\alpha}}{\alpha \sum_{i=1}^{n} f(d_i) v_i^{\alpha}}$$

Logit demand. Differentiating profit Equation 4.9 and substituting c_i for $\gamma f(d_i)$, we can express γ:

$$\gamma = \frac{\sum \left(e^{\alpha(v_i - P_0)} \left(\alpha P_0 - 1 - \sum_{i=1}^{n} e^{\alpha(v_i - P_0)} \right) \right)}{\alpha \sum_{i=1}^{n} f(d_i) e^{\alpha(v_i - P_0)}}$$

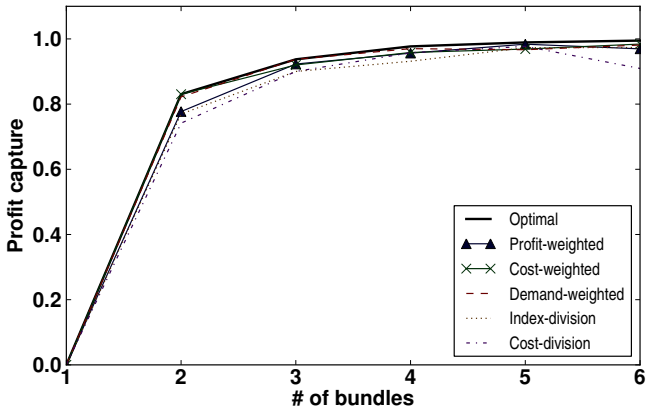
4.3.2 How Should Tiers Be Structured?

ISPs must judiciously choose how they bundle traffic flows into tiers. As shown in Section 4.1.1, today's ISPs often offer at most two or three bundles with different prices. We define six bundling strategies that classify and group traffic flows according to their cost, demand, or potential profit to the ISP. We then evaluate them and show that, assuming the right bundling strategy is used, ISPs typically need only a few bundles to collect near-optimal profit.

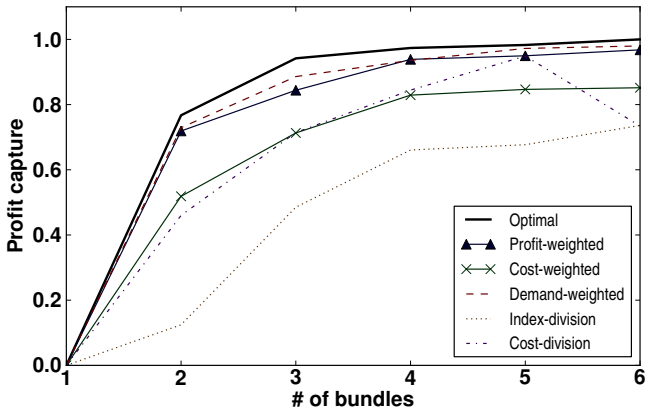
4.3.2.1 Bundling strategies

Optimal. We exhaustively search all possible combinations of bundles to find the one that yields the most profit. This approach gives optimal results and also serves as our *baseline* against which we compare other strategies. Computing the optimal bundling is computationally expensive: for example, there is more than a billion ways to divide one hundred traffic flows into six pricing bundles. Presented below, all of the other bundling strategies employ heuristics to make bundling computationally tractable.

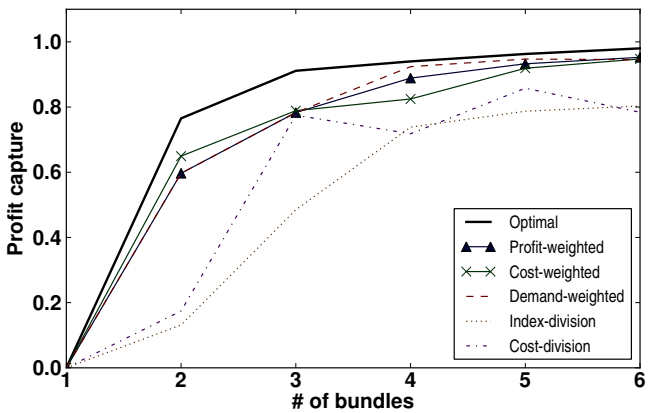
Demand-weighted. In this strategy, we use an algorithm inspired by token buckets to group traffic flows to bundles. First, we set the overall token budget as the sum of the original demand of all flows: $T = \sum_i q_i$. Then, for each bundle j we assign the same token budget $t_j = T/B$, where B is the number of bundles we want to create. We sort the flows in decreasing order of their demand and traverse them one-by-one. When traversing flow i, we assign it to the first bundle j that either has no flows assigned to it or has a budget $t_j > 0$. We reduce the budget of that bundle by q_i. If the resulting budget $t_j < 0$, we set



(a) European ISP.

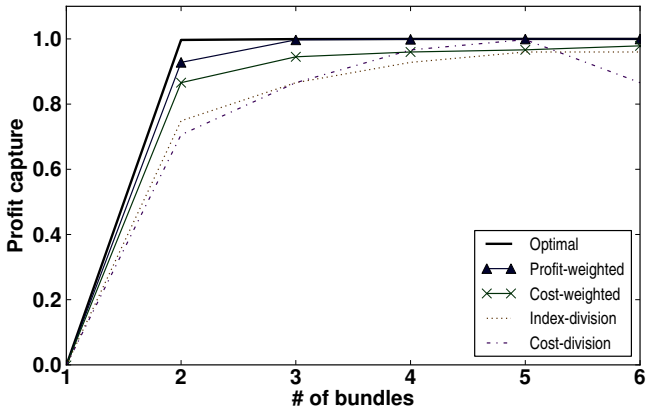


(b) Internet2.

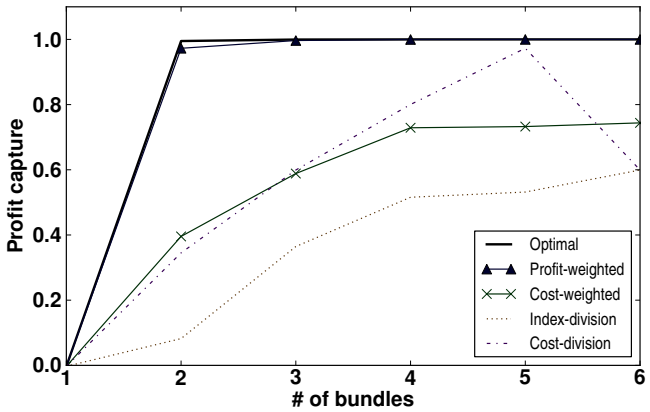


(c) International CDN.

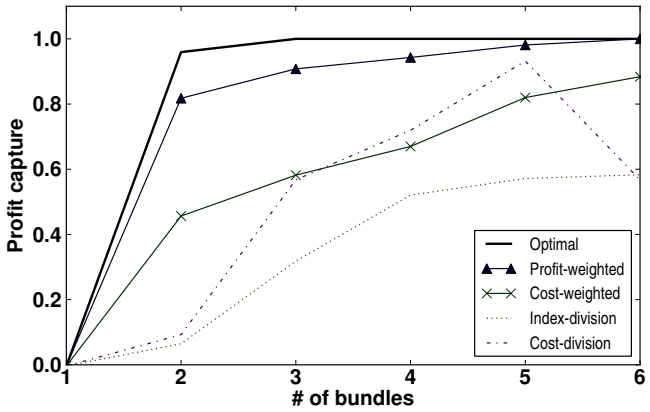
Figure 4.8: Profit capture for different bundling strategies in constant elasticity demand.



(a) European ISP.



(b) Internet2.



(c) International CDN.

Figure 4.9: Profit capture for different bundling strategies in logit demand.

$t_{j+1} = t_{j+1} + t_j$. After traversing all the flows, the token budget of every bundle will be zero, and each flow will be assigned to a bundle. The algorithm leads to separate bundles for high demand flows and shared bundles for low demand flows. For example, if we need to divide four flows with demands 30, 10, 10, and 10 into two bundles, the algorithm will place the first flow in the first bundle, and the other three flows in the second bundle.

Cost-weighted. We use the same approach as in demand-weighted bundling, but we set the token budget to $T = \sum_i 1/c_i$. When placing a flow in a bundle we remove a number of tokens equal to the inverse of its cost. This approach creates separate bundles for local flows and shared bundles for flows traversing longer distances. The current ISP practices of offering *regional pricing* and *backplane peering* maps closely to using just two or three bundles arranged using this cost-weighted strategy.

Profit-weighted. The bundling algorithms described above consider cost and demand separately. To account for cost and demand together, we estimate *potential profit* each flow could bring. We use the potential profit metric to apply the same weighting algorithm as in cost and demand-weighted bundling. In case of constant elasticity demand, we derive potential profit of each flow i:

$$\pi_i = \frac{v_i^\alpha}{\alpha} \left(\frac{\alpha c_i}{\alpha - 1} \right)^{1-\alpha} \tag{4.14}$$

For the logit demand, substituting p_i in Equation 4.10 yields:

$$\pi_i = K s_i (p_i - c_i) = \frac{K s_i}{\alpha s_0} \propto q_i \tag{4.15}$$

Cost division. We find the most expensive flow and divide the cost into ranges according to that value. For example, if we want to introduce two bundles and the most expensive flow costs \$10/Mbps/month to reach, we assign flows that cost \$0–\$4.99 to the first bundle and flows that cost \$5–\$10 to the second bundle.

Index division. Index-division bundling is similar to cost division bundling, except that we rank flows according to their cost and use the rank, rather than the cost, to perform the division into bundles.

4.3.2.2 The effects of different bundling strategies

To evaluate the bundling strategies described above, we compute the profit-maximizing prices and measure the resulting pricing outcome in terms of *profit capture*. Profit capture indicates what fraction of the maximum possible profit—the profit attained using an infinite number of bundles—the strategy captures. For example, if the maximum attainable profit is 30% higher than the original profit, while the profit from using two bundles is 15% higher than the original profit, the profit capture with two bundles attains 0.5 of profit capture. Formally, profit capture is $(\pi_{\mathsf{new}} - \pi_{\mathsf{original}})/(\pi_{\mathsf{max}} - \pi_{\mathsf{original}})$.

Figures 4.8 and 4.9 show the profit capture for different bundling strategies, across the three data sets, while varying the number of bundles. For the results shown here, we use both the constant elasticity and the logit demand models and the linear cost model. We set the price sensitivity α to 1.1, the original, blended rate P_0 to \$20, the cost tuning parameter θ to 0.2, and the original market fraction that sends no traffic s_0 to 0.2. We explore the effect of varying these parameters in Section 4.3.4.

Optimal versus heuristics-based bundling. With an appropriate bundling strategy, the ISP attains maximum profit with just 3–4 bundles. As expected, the optimal flow bundling strategy captures the most profit for a given number of bundles. We observe that the EU ISP captures more profit with two bundles than other networks. We attribute this effect to the low coefficient of variation (CV) of demand to different destinations, which limits the benefits of having more pricing bundles. We also discover that, given fixed demand, a high CV of distance (cost) leads to higher absolute profits. With only minor exceptions, the profit-weighted bundling heuristic is almost as good as the the optimal bundling, followed by the cost-weighted bundling heuristic. Deeper analysis, beyond the scope of this work,

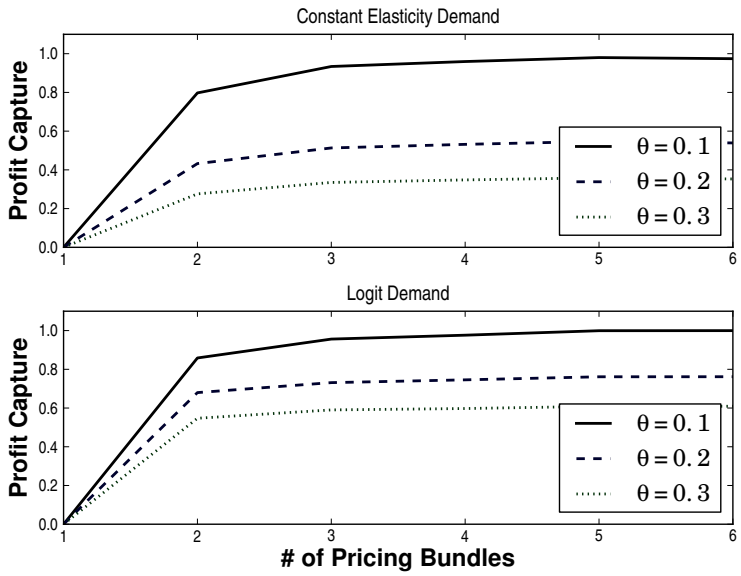


Figure 4.10: Profit increase in EU ISP network using linear cost model.

could show what specific input data conditions cause the profit-weighted flow bundling heuristic to produce bundlings superior to the cost-weighted heuristic.

Logit profit capture. Maximum profit capture occurs more quickly in the logit model because (1) the total demand (including s_0 option) is constant, and (2) the model is sensitive to differences in valuation of different flows. When there is a flow with a significantly higher difference between valuation and cost ($v_i - c_i$), it absorbs most of the demand. In this model, with just two pricing tiers, local and non-local traffic are separated into distinct bundles that closely represent the *backplane peering* and *regional pricing* for local area service models.

4.3.3 Consumer Surplus Analysis

The consumer surplus gain shown in Figure 4.12 is normalized to the surplus gain the consumers get when ISPs maximize their profit with an infinite number of pricing tiers. As expected, the surplus gain follows closely, if not precisely, ISP profit gains.

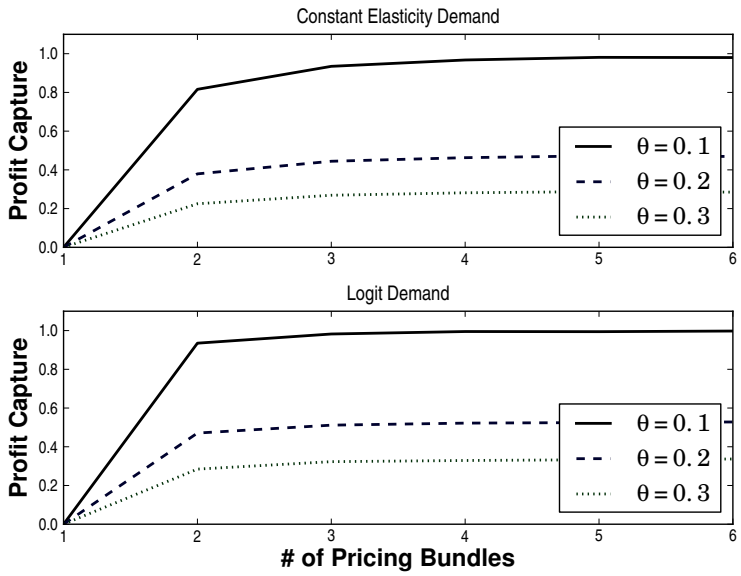


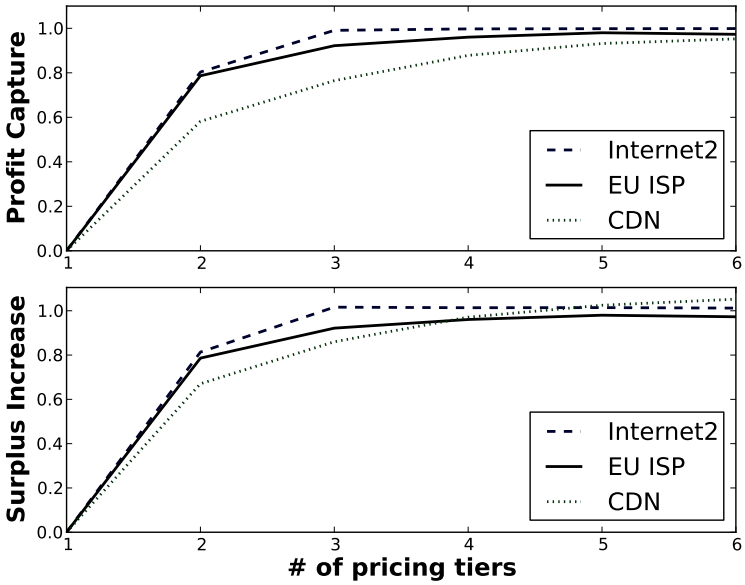
Figure 4.11: Profit increase in EU ISP network using concave cost model.

4.3.4 Sensitivity Analysis

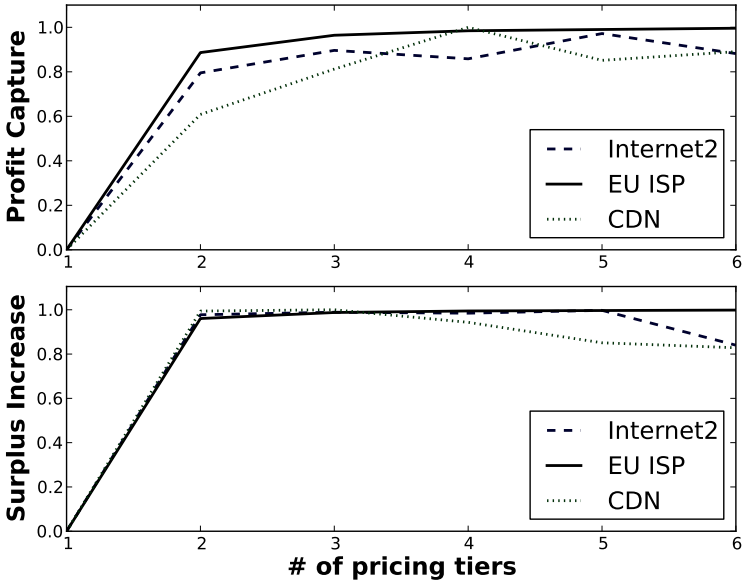
We explore the robustness of our results to cost models and input parameter settings. As we vary an input parameter under test, other parameters remain constant. Unless otherwise noted, we use profit-weighted bundling, the EU ISP dataset, sensitivity $\alpha = 1.1$, the linear cost model with base cost $\theta = 0.2$, blended rate $P_0 = \$20.0$, and, in the logit model, $s_0 = 0.2$ (the original market fraction that sends no traffic).

4.3.4.1 Effects of cost models

We aim to see how cost models and settings within these models qualitatively affect our results from the previous section. We show how profit changes as we increase the number of bundles for different settings of the cost model parameters (θ), described in Section 4.2.3. We find that for different θ settings most of the attainable profit is still captured in 2-3 bundles. Unlike in other sections, in Figures 4.10–4.14, we normalize the profit of all the plots in the graphs to the highest observed profit. In other words, π_{max} in these figures is not the maximum profit of each plot, but the maximum profit of the plot with highest profit in the figure. Normalizing by the highest observed profit allows us to show how changing



(a) Constant elasticity demand.



(b) Logit.

Figure 4.12: ISP profit and consumer surplus change as ISP employ an increasing number of pricing tiers. Concave cost model, $\theta = 0.2$, $\alpha = 1.1$, $s_0 = 0.2$.

the parameter θ affects the amount of profit that the ISP can capture.

Linear cost. Figure 4.10 shows profit increase in the EU ISP network as we vary the number of bundles for different settings of θ. As expected, most of the profit is still attained with 2–3 pricing bundles. We also observe that the increase in the base cost (θ) causes a decline in the maximum attainable profit. The reduction in maximum attainable profit is expected, as increasing the base cost reduces the coefficient of variation (CV) of the cost of different flows and thus reduces the opportunities for variable pricing and profit capture. We can also see, as shown in previous section, that the logit demand model attains more profit than the constant elasticity demand model with the same number of pricing bundles.

Concave cost. Figure 4.11 shows the profit increase as we vary the number of bundles for different settings of θ for the concave cost model. The observations and results are similar to the linear cost model, with one notable exception. The amount of profit the ISP can capture decreases more quickly in the concave cost model than in the linear cost model for the same change in the base-cost parameter θ. This is due to the lower CV of cost in the concave model than in the linear cost model. In other words, applying the log function on distance (as described in Section 4.2.3) reduces the relative cost difference between flows traveling to local and remote destinations.

Regional cost. In the regional cost model, the parameter θ is an exponent which adjusts the price difference between three different regions: local, national, and international. Figure 4.13 shows the profit increase in the EU ISP network as we vary number of bundles for different settings of θ. Higher θ values result in a higher CV of cost in different regions which, in turn, in both demand models produces higher profit. Using constant elasticity demand we observe a small dip in profit when using five and six bundles, which recovers later with more bundles. Such dips are expected when there are only a few traffic classes. For example, if traffic had just two distinct cost classes, two judiciously selected bundles

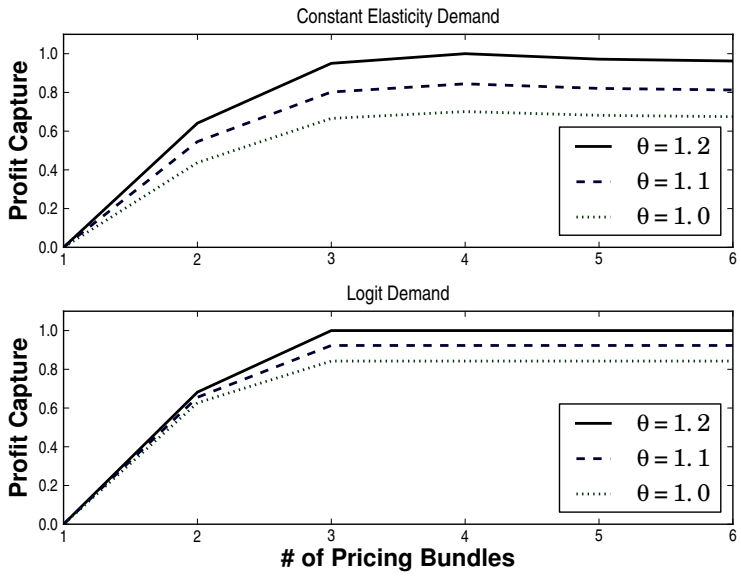


Figure 4.13: Profit increase in the EU ISP network using regional cost model.

could capture most of the profit. Adding a third bundle can reduce the profit if that third bundle contains flows from both of the classes (as may happen in a suboptimal bundling).

Destination type-based cost. Destination type-based cost model emulates "on-net" and "off-net" types of traffic in an ISP network. As described in Section 4.2.3, we assume that "on-net" traffic costs less than "off-net" traffic. We vary θ, which represents a fraction of "on-net" traffic in each flow. The standard profit-weighting algorithm does not work well with the destination type-based cost model. The effect observed in the regional cost model—where five bundles produce slightly lower profit then four bundles—is more pronounced when we have just two distinct flow classes. One heuristic that works reasonably well is as follows: we update the profit-weighting heuristic to never group traffic from two different classes into the same bundle. Figure 4.14 shows how profit increases with an increasing number of bundles. Since there are two major classes of traffic ("on-" and "off-net"), most profit is attained with two bundles for both demand models. In this cost model, as in other cost models, the same change in CV of cost (induced by the parameter θ) causes a greater change in profit capture for constant elasticity demand than for logit

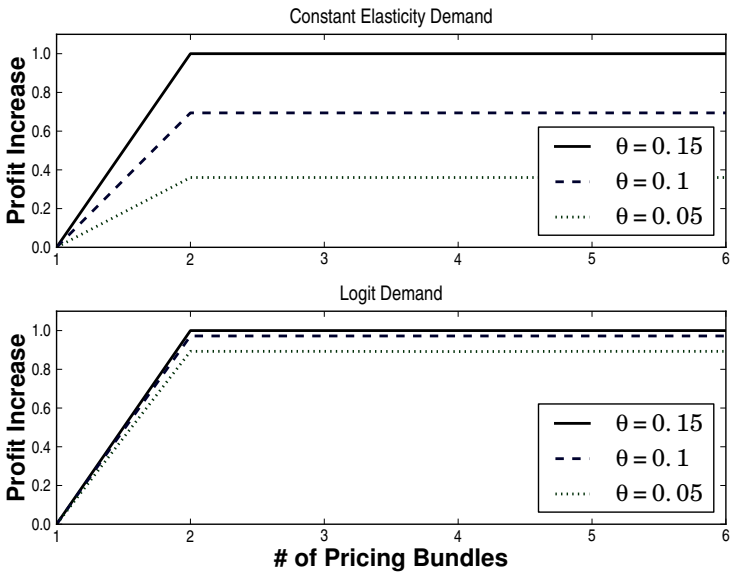


Figure 4.14: Profit increase in the EU ISP network using destination type cost model.

demand.

4.3.4.2 Sensitivity to parameter settings

The models we use rely on a set of parameters, such as price sensitivity (α), price of the original bundle (P_0), and, in the logit model, the share of the market that corresponds to deciding not to purchase bandwidth (s_0). In this section, we analyze how sensitive the model is to the choice of these parameters.

Figures 4.15–4.17 show how profit capture is affected by varying price sensitivity α, blended rate P_0, and non-buying market share s_0, respectively. Each data point in the figures is obtained by varying each parameter over a range of values. We vary α between 1 and 10, P_0 between 5 and 30, and s_0 between 0 and 0.9. As we vary the parameters, we select and plot the *minimum* observed profit capture over the whole parameter range, for the profit-weighted strategy with different numbers of bundles. In other words, these plots show the worst case relative profit capture for the ISP over a range of parameter values. The trend of these minimum profit capture points is qualitatively similar to patterns in Figures 4.8 and 4.9. For example, using the CED model and grouping flows in two bundles in the

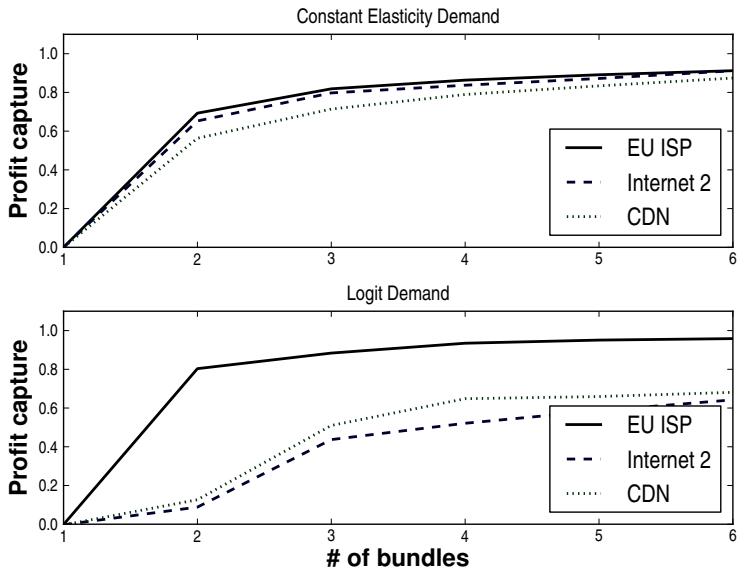


Figure 4.15: Minimum profit capture for a fixed number of bundles over a range of α between 1 and 10.

EU ISP yields around 0.8 profit capture, regardless of price sensitivity, blending rate, and market share. These results indicate that our model is robust to a wide range of parameter values.

4.4 Implementing Tiered Pricing

ISPs can implement the type of tiered pricing that we describe in Section 4.3 without any changes to their existing protocols or infrastructure, and ISPs may already be using the techniques we describe below. If that is the case, they could simply apply a profit-weighted bundling strategy to re-factor their pricing to improve their profit, possibly without even making many changes to the network configuration. We describe two tasks associated with tiered pricing: associating flows with tiers and accounting for the amount of traffic the customer sends in each tier.

An ISP performs blended-rate pricing over a single link. When a link is used for transit, the upstream Internet service provider typically announces a complete Internet routing

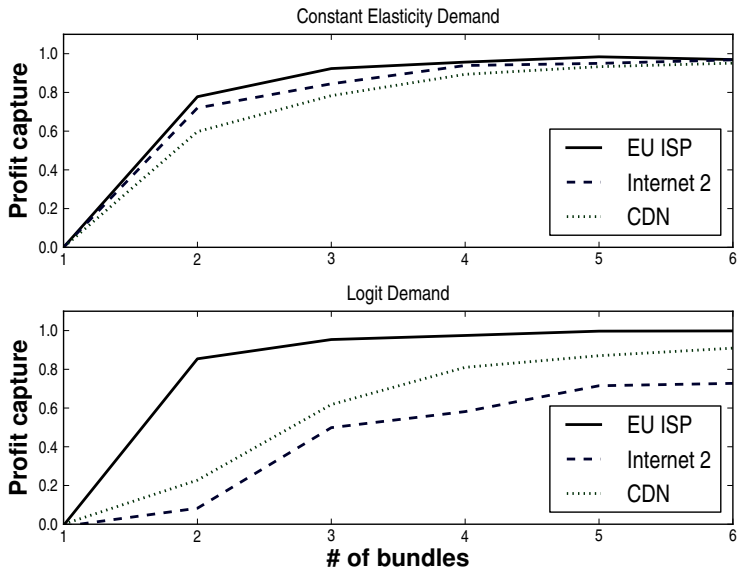


Figure 4.16: Minimum profit capture for a fixed number of bundles over a range of starting prices $P_0 \in [5, 30]$.

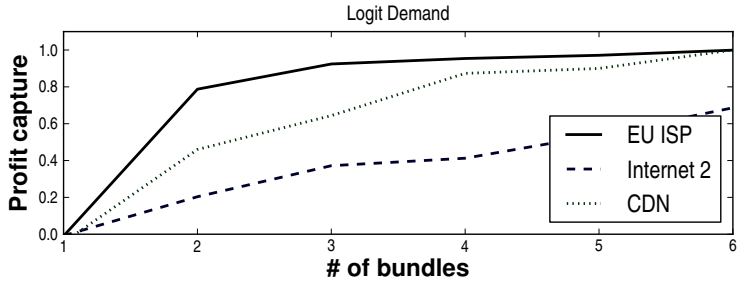


Figure 4.17: Maximum profit capture for a fixed number of bundles over a range of fractions of users who decide not to participate in the market $s_0 \in (0, 1)$.

table over a single session. A Simple Network Management Protocol (SNMP) [59] station polls the interface associated with a link to a customer every five minutes and records the bytes transferred for each five-minute interval in an accounting database, as shown in Figure 4.18. The provider ISP then implements 95th percentile pricing by charging the customer a *blended rate* based on the traffic load in 95th percentile of all samples.

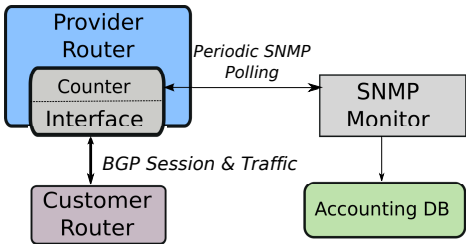


Figure 4.18: SNMP-based accounting for blended rate pricing.

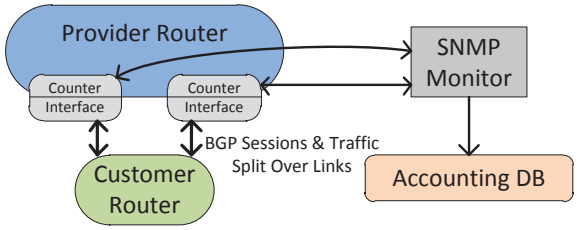
4.4.1 Associating Flows with Tiers

Associating each flow (or destination) with a tier can be done within the context of today's routing protocols. When the upstream ISP sends routes to its customer, it can "tag" routes it announces with a label that indicates which tier the route should be associated with; ISPs can use BGP extended communities to perform this tagging. Because the communities propagate with the route, the customer can establish routing policies on every router within its own network based on these tags.

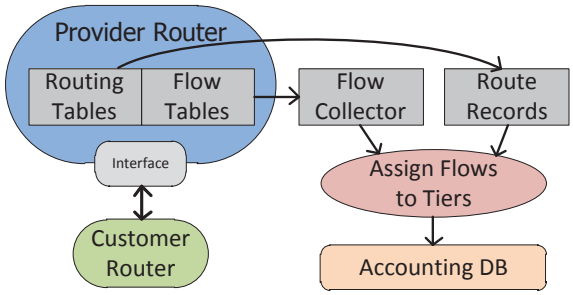
Suppose that a large transit service provider has routers in different geographic regions. Routers at an exchange point in, say, New York, might advertise routes that it learned in Europe with a special tag indicating that the path the route takes is trans-Atlantic and, hence, bears a higher price than other, regional routes. The customer can then use the tag to make routing decisions. For example, if a route is tagged as an expensive long-distance route, the customer might choose to use its own backbone to get closer to destination instead of performing the default "hot-potato" routing (*i.e.*, offloading the traffic to a transit network as quickly as possible). A large customer might also use this pricing information to better plan its own network growth.

4.4.2 Accounting

Implementing tiered pricing requires accounting for serviced traffic either on a *per-link* or *per-flow* basis.



(a) SNMP-Based Accounting.



(b) Flow-Based Accounting.

Figure 4.19: Implementing accounting for tiered pricing.

Link-Based Accounting. As shown in Figure 19(a), an edge router can establish two or more physical or virtual links to the customer, with a Border Gateway Protocol (BGP) [95] session for each physical or virtual link. In this setup, each pricing tier would have a separate link. Each link carries the traffic only to the set of destinations advertised over that session (*e.g.*, on-net traffic, backplane peering traffic). Because each link has a separate routing session and only exchanges routes associated with that pricing tier, the customer and provider can ensure that traffic for each tier flows over the appropriate link: The customer knows exactly which traffic falls into which pricing tier based on the session onto which it sends traffic. Billing may also be simpler and easier to understand, since, in this mode, a provider can simply bill each link at a different rate. Unfortunately, the overhead of this accounting method grows significantly with the number of pricing levels ISP intends to support.

Flow-Based Accounting. In flow-based accounting, as in traditional peering and transit, an upstream ISP and a customer establish a link with a single routing session. As

shown in Figure 19(b), the accounting system collects both flow statistics (*e.g.*, using NetFlow [82]) and routing information to determine resource usage. For the purposes of accounting, bundling effectively occurs after the fact: flows can be mapped to distances using the routing table information and priced accordingly, exactly as we did in our evaluation in Section 4.3. Assuming flow and routing information collection infrastructure in place, flow-based accounting may be easier to manage, and it is easier to bundle flows into different bins according to pre-agreed bundling scheme.

4.5 Related Work

Developing and analyzing pricing models for the Internet is well-researched in both networking and economics. Two aspects are most relevant for our work: the unbundling of connectivity and the dimensions along which to unbundle it. Although similar studies of pricing exist, none have been evaluated in the context of wholesale Internet transit and real network demand and topology data.

The unbundling of connectivity services refers to the setting of different prices for such services along various usage dimensions such as volume, time, destination, or application type. Seminal works by Arrow and Debreu [22] and McKenzie [77] show that markets where commodities are sold at infinitely small granularities are more efficient. More recent studies however, demonstrate that unbundling may be inefficient in certain settings, such as when selling information goods with zero or very low marginal cost (such as access to online information) [23, 70, 81]. This is not always the case with the connectivity market, where ISPs incur different costs to deliver traffic to different destinations. In addition, many service providers already use price discrimination [86].

Kesidis *et al.* [67] and Shakkottai *et al.* [101] study the benefits of pricing connectivity based on volume usage and argue that, with price differentiation, one can use resources more efficiently. In particular, Kesidis *et al.* show that usage-based unbundling may be

even more beneficial to access networks rather than core networks. Time is another dimension along which providers can unbundle connectivity. Jiang *et al.* [65] study the role of time preference in network prices and show analytically that service providers can achieve maximum revenue and social welfare if they differentiate prices across users and time. Hande *et al.* [58] characterize the economic loss due to ISP inability or unwillingness to price broadband access based on time of day.

4.6 Summary

As the price of Internet transit drops, transit providers are selling connectivity using "tiered" contracts based on traffic cost, volume, or destination to maintain profits. We have studied two questions: How does tiered pricing benefit both ISPs and their customers? and How should ISPs structure the connectivity tiers they sell to maximize their profits? We developed a model for an Internet transit market that helps ISPs evaluate how they should arrange traffic into different tiers, and how they should set prices for each of those tiers. We have applied our model to traffic demand and topology data from three large ISPs to evaluate various bundling strategies.

We find that the common ISP practice of structuring tiered contracts according to the cost of carrying the traffic flows (*e.g.*, offering a discount for traffic that is local) is suboptimal. Dividing the contract into only three or four tiers based on *both traffic cost and demand* yields near-optimal profit for the ISP; other strategies such as cost division bundling also work well. We also find that networks with primarily lower cost traffic (either local or traveling short distances) require fewer tiers to extract maximum profit than other networks do.

CHAPTER V

CONCLUDING REMARKS

As the Internet growth is accelerating, so does the diversity of applications that use it. Just in last few decades we have seen dramatic shifts in use of the Internet. Starting merely as a platform to send emails, the Internet embraced peer-to-peer networks, search engines, online productivity tools, and video-on-demand applications. Many of such services have different requirements for the Internet connectivity. For example, a video-on-demand service might prefer cheap and bulky Internet paths, while search providers would pay premium for lower latencies between them and the end-users.

5.1 Towards Increasing Granularity of Wide-Area Route Control

Although different online services might have different networking requirements, not all of them have access to wide-area route control which would allow them to choose different Internet paths. Specifically, the services hosted at the facilities of cloud computing providers are at the mercy of routing policies of said providers. For example, an Amazon EC2 data center might hosts thousands of online service, yet each of these services uses the same path (selected by Amazon edge router) to reach the end users. In this dissertation, I put forward a thesis that many online services would benefit from a better access to a granular wide-area route control.

Given the evolving nature of Internet online services, the current ISPs business models also deserve scrutiny. Currently, the wholesale Internet transit is sold at bulk, using so called "blended rates". While such one-size-fits-all pricing works fine when all services are similar, this pricing approach might not be the most effective for services with diverse Internet uses. For example, some services, such as online gaming, reach a global audience, while other services, such as content streaming, usually are constrained to a single locale.

These two services will likely incur different cost to ISP, even when sending the same amount of data. In this dissertation, I model and analyse the ISP pricing strategies as it pertains to such online service diversity.

5.2 Summary of Contributions

My thesis has three claims: 1) we can make access to wide-area routing easier, 2) granular wide-area routing can significantly benefit the online services, and 3) the business models employed by today's ISPs are adequate to support increasing diversity of online services. My contributions to each of these claims are as follows:

1. **Transit Portal platform.** I propose, design, build, and deploy a testbed of a Transit Portal (TP) platform. TP provides online services hosted in cloud computing environment with an illusion of direct access to upstream ISPs. The clients of such platform can enjoy the same level of route control as their cloud hosting provider. Our TP testbed consists of five nodes and provides researchers with access to the Internet route control.
2. **Fine-grained routing benefit analysis.** I show that services that are replicated geographically, can still benefit significantly if they also employ a fine-grained routing. Specifically, using our Transit Portal testbed, I show that by performing joint content and network routing, OSPs can achieve 22% larger latency reduction than can be obtained by content routing alone.
3. **ISP profit and consumer surplus modeling.** In the dissertation, I perform extensive modeling of ISP pricing strategies, as it pertains to ISP profits and consumer surplus. I build and evaluate a pricing model with two different demand models and four different cost models. I evaluate the models on three different real-world data sets. My results show that ISPs gain little if they price services at granularities exceeding two to three tiers. In other words, I show that the current ISP pricing models are adequate to support increasing online service diversity.

5.3 Future Directions

Although I consider the dissertation complete, there are ample areas for further exploration. In all of my dissertation I have relied only on the existing routing protocols to provide increased routing diversity to the online services and to enhance the performance they deliver to the end users. The existing routing protocols, however, namely BGP, have numerous known deficiencies and, in the long run, might be superseded. More specifically, industry practitioners are considering Locator/ID Separator Protocol (LISP) as a future replacement for BGP and it warrants more attention from the research community.

Whatever is the protocol that drives the future Internet, the paths provided by such protocol have to be evaluated and selected. The challenge today is that the online services are left on their own to evaluate the end-to-end performance to their users. Interestingly, many of such online service do these measurements concurrently and without coordination or information sharing. For example, Amazon might be monitoring quality of paths to its data centers, while at the same time many services inside the Amazon data centers to the same. I foresee a need for a mechanism to share such performance information between cloud providers and the hosted services so that hosted services can make more informed route choices.

Finally, the last part of my dissertation took on modeling the ISP business models. No model is perfect and mine certainly can be improved. What's more, a more holistic approach is necessary to model the Internet market. Such an approach should not only look at the interaction between an ISP and its clients, but also look at the evolution of all of the ISPs connection and the place of each ISP in the Internet ecosystem.

Bibliography

[1] "BGPlay Route Views." http://bgplay.routeviews.org/bgplay/.

[2] "node.js." http://nodejs.org.

[3] "PECAN measurement dataset." https://sites.google.com/site/pecanrouting/.

[4] "SPDY: An experimental protocol for a faster web." http://www.chromium.org/spdy/spdy-whitepaper.

[5] "Gaikai Demo." http://www.dperry.com/archives/news/dp_blog/gaikai_-_video/, 2010.

[6] "ooVoo." http://www.oovoo.com/, 2010.

[7] "Skype." http://www.skype.com/, 2010.

[8] "Slice-based facility architecture." http://www.cs.princeton.edu/~llp/arch_abridged.pdf, 2010.

[9] "Vytautas Valancius research web page." http://valas.gtnoise.net, 2011.

[10] "World Bank: World Development Indicators." http://data.worldbank.org/data-catalog/world-development-indicators, 2011.

[11] "Chrome software updates: Courgette." http://dev.chromium.org/developers/design-documents/software-updates-courgette, 2012.

[12] "Transit Portal web console." http://tp.gtnoise.net, 2012.

[13] ADAM INTERNET, "Unmetered content." http://www.adam.com.au/unmetered/unmetered.php. Retrieved: June 2011.

[14] AGARWAL, S., DUNAGAN, J., JAIN, N., SAROIU, S., WOLMAN, A., and BHOGAN, H., "Volley: Automated data placement for geo-distributed cloud services," in *Proc. 7th USENIX NSDI*, (San Jose, CA), Apr. 2010.

[15] AKAMAI, "http://www.akamai.com," 2010.

[16] AKELLA, A., MAGGS, B., SESHAN, S., and SHAIKH, A., "On the performance benefits of multihoming route control," *IEEE/ACM Transactions on Networking*, vol. 16, Feb. 2008.

[17] AKELLA, A., MAGGS, B., SESHAN, S., SHAIKH, A., and SITARAMAN, R., "A measurement-based analysis of multihoming," in *Proc. ACM SIGCOMM*, (Karlsruhe, Germany), Aug. 2003.

[18] AKELLA, A., PANG, J., MAGGS, B., SESHAN, S., and SHAIKH, A., "A comparison of overlay routing and multihoming route control," in *Proc. ACM SIGCOMM*, (Portland, OR), Aug. 2004.

[19] AKELLA, A., SESHAN, S., and SHAIKH, A., "Multihoming performance benefits: An experimental evaluation of practical enterprise strategies," in *Proc. USENIX Annual Technical Conference*, (Boston, MA), June 2004.

[20] ANDERSEN, D. G., BALAKRISHNAN, H., KAASHOEK, M. F., and MORRIS, R., "Resilient Overlay Networks," in *Proc. 18th ACM Symposium on Operating Systems Principles (SOSP)*, (Banff, Canada), Oct. 2001.

[21] ARMBRUST, M., FOX, A., GRIFTH, R., JOSEPH, A. D., KATZ, R., KONWINSKI, A., LEE, G., PATTERSON, D., RABKIN, A., STOICA, I., and ZAHARIA, M., "Above the clouds: A Berkeley view of cloud computing," tech. rep., University of California at Berkeley, Feb. 2009.

[22] ARROW, K. J. and DEBREU, G., "Existence of equilibrium for competitive economy," *Econometrica*, vol. 22, no. 3, pp. 265–290, 1954.

[23] BAKOS, Y. and BRONJOLFSSON, E., "Bundling and competition on the Internet," *Marketing Science*, vol. 19, Jan. 1998.

[24] BESANKO, D., GUPTA, S., and JAIN, D., "Logit demand estimation under competitive pricing behavior: An equilibrium framework," *Management Science*, vol. 44, Nov. 1998.

[25] " bgpsimple: simple BGP peering and route injection script ." `http://code.google.com/p/bgpsimple/`.

[26] BHATTACHARJEE, S., AMMAR, M. H., ZEGURA, E. W., SHAH, V., and FEI, Z., "Application layer anycasting," in *Proc. IEEE INFOCOM*, (Kobe, Japan), Apr. 1997.

[27] BNSL, "Tariff for leased lines." `http://www.bsnl.co.in/service/2mbps.pdf`. Retrieved: June 2011.

[28] BROWN, M. A., POPESCU, A., and ZMIJEWSKI, E., "Peering Wars: Lessons Learned from the Cogent-Telia De-peering," in *NANOG 43*, June 2008.

[29] BRUCE, M. A., SHEPHERD, B., SRINIVASAN, A., WINKLER, P., and ZANE, F., "Clustering and server selection using passive monitoring," in *Proc. IEEE INFOCOM*, (New York, NY), June 2002.

[30] BUSH, R., MAENNEL, O., ROUGHAN, M., and UHLIG, S., "Internet optometry: Assessing the broken glasses in internet reachability," in *Proc. Internet Measurement Conference*, (Chicago, Illinois), Oct. 2009.

[31] CALLAHAN, T., ALLMAN, M., and PAXSON, V., "A longitudinal view of http traffic," in *Passive & Active Measurement (PAM)*, (Zurich, Switzerland), Apr. 2010.

[32] CHANG, H., JAMIN, S., and WILLINGER, W., "To peer or not to peer: Modeling the evolution of the Internet's AS-level topology," in *Proc. IEEE INFOCOM*, (Barcelona, Spain), Mar. 2006.

[33] CHENG, Y. and OTHERS, *TCP Fast Open*. Internet Engineering Task Force, Sept. 2011. http://www.ietf.org/id/draft-cheng-tcpm-fastopen-00.txt.

[34] CHU, J. and OTHERS, *Increasing TCP's Initial Window*. Internet Engineering Task Force, Oct. 2011. http://tools.ietf.org/html/draft-ietf-tcpm-initcwnd-01.

[35] CHUNGHWA TELECOM, "Leased line pricelist." http://www.cht.com.tw/CHTFinalE/Web/Business.php?CatID=476. Retrieved: June 2011.

[36] CISCO, "SFP optics for gigabit ethernet applications." http://www.cisco.com/en/US/prod/collateral/modules/ps5455/ps6577/product_data_sheet0900aecd8033f885.html. Retrieved: June 2011.

[37] "Cisco Optimized Edge Routing (OER)." http://www.cisco.com/en/US/products/ps6628/products_ios_protocol_option_home.html, 2010.

[38] "Cisco Visual Networking Index." www.cisco.com/web/go/vni.

[39] "Cogent Communications BGP Communities." http://www.onesc.net/communities/as174/, 2010.

[40] DHAMDHERE, A. and DOVROLIS, C., "ISP and egress path selection for multihomed networks," in *Proc. IEEE INFOCOM*, Apr. 2006.

[41] DHAMDHERE, A., FRANCOIS, P., and DOVROLIS, C., "A value based framework for internet peering agreements," in *Proc. International Teletraffic Congress (ITC)*, 2010.

[42] "Emulab." http://www.emulab.net/.

[43] "Equinix Direct." http://www.equinix.com/solutions/connectivity/equinixdirect/, 2010.

[44] ETISALAT ISP, "Leased circuit rental charges." http://tinyurl.com/66tfvj6. Retrieved: June 2011.

[45] FARINACCI, D., FULLER, V., ORAN, D., and MEYER, D., *Locator/ID Separation Protocol (LISP)*. Internet Engineering Task Force, Apr. 2008. Internet Draft (http://tools.ietf.org/html/draft-farinacci-lisp-07). Work in progress, expires October 2008.

[46] FEAMSTER, N. and BALAKRISHNAN, H., "Verifying the correctness of wide-area Internet routing," Tech. Rep. MIT-LCS-TR-948, Massachusetts Institute of Technology, May 2004.

[47] FEAMSTER, N., BORKENHAGEN, J., and REXFORD, J., "Guidelines for interdomain traffic engineering," *ACM Computer Communications Review*, vol. 33, Oct. 2003.

[48] FEAMSTER, N., MAO, Z. M., and REXFORD, J., "BorderGuard: Detecting cold potatoes from peers," in *Proc. Internet Measurement Conference*, (Taormina, Italy), Oct. 2004.

[49] GEELAN, J., "Twenty one experts define cloud computing.," Aug. 2008. `http://virtualization.sys-con.com/node/612375`.

[50] "GENI: Global Environment for Network Innovations." `http://www.geni.net/`.

[51] GODFREY, B., GANICHEV, I., SHENKER, S., and STOICA, I., "Pathlet routing," in *Proc. ACM SIGCOMM*, (Barcelona, Spain), Aug. 2009.

[52] GOEL, A., KRASIC, C., and WALPOLE, J., "Low-latency adaptive streaming over TCP," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 4, Aug. 2008.

[53] GOLDENBERG, D. K., QIU, L., XIE, H., YANG, Y. R., and ZHANG, Y., "Optimizing cost and performance for multihoming," in *Proc. ACM SIGCOMM*, (Portland, OR), Aug. 2004.

[54] GOVINDAN, R., ALAETTINOGLU, C., VARADHAN, K., and ESTRIN, D., "Route Servers for Inter-Domain Routing," *Computer Networks and ISDN Systems*, vol. 30, pp. 1157–1174, 1998.

[55] GUAVUS, "Profitable tiered pricing." `http://www.guavus.com/solutions/tiered-pricing`. Retrieved: June 2011.

[56] GUMMADI, K. P., MADHYASTHA, H. V., GRIBBLE, S. D., LEVY, H. M., and WETHERALL, D., "Improving the reliability of Internet paths with one-hop source routing," in *Proc. 6th USENIX OSDI*, (San Francisco, CA), Dec. 2004.

[57] GUO, F., CHEN, J., LI, W., and CKER CHIUEH, T., "Experiences in building a multihoming load balancing system," in *In Proceedings of IEEE INFOCOM, Hong Kong*, 2004.

[58] HANDE, P., CHIANG, M., CALDERBANK, R., and ZHANG, J., "Pricing under constraints in access networks: Revenue maximization and congestion management," in *Proc. IEEE INFOCOM*, (San Diego, CA), Mar. 2010.

[59] HARRINGTON, D., PRESHUN, R., and WIJNEN, B., *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*. Internet Engineering Task Force, Dec. 2002. RFC 3411.

[60] HUANG, C., HOLT, N., WANG, A., GREENBERG, A., JIN LI, and ROSS, K. W., "A DNS reflection method for global traffic management," in *Proc. USENIX Annual Technical Conference*, (Boston, MA), June 2010.

[61] HUANG, C., MALTZ, D. A., GREENBERG, A., and LI, J., "Public DNS system and global traffic management," in *Proc. IEEE INFOCOM*, (Shanghai, China), Apr. 2011.

[62] "Internap." `http://www.internap.com/`, 2009.

[63] "ITU Telecommunication Indicator Handbook." `http://www.itu.int/ITU-D/ict/publications/world/material/handbook.html`. Retrieved: June 2011.

[64] "V8 JavaScript engine." `http://code.google.com/p/v8/`.

[65] JIANG, L., PAREKH, S., and WALRAND, J., "Time-dependent network pricing and bandwidth trading," in *Proc. IEEE BoD*, 2008.

[66] JOHARI, R. and TSITSIKLIS, J., "Routing and Peering in a Competitive Internet," Tech. Rep. LIDS Publication 2570, Massachusetts Institute of Technology, 2003. `http://www.stanford.edu/~rjohari/pubs/netgamepaper.pdf`.

[67] KESIDIS, G., DAS, A., and VECIANA, G. D., "On flat-rate and usage-based pricing for tiered commodity Internet services," in *Proc. CISS*, 2008.

[68] KRISHNAN, R., MADHYASTHA, H. V., JAIN, S., SRINIVASAN, S., KRISHNAMURTHY, A., ANDERSON, T., and GAO, J., "Moving beyond end-to-end path information to optimize CDN performance," in *Proc. Internet Measurement Conference*, 2009.

[69] LABOVITZ, C., IEKEL-JOHNSON, S., MCPHERSON, D., OBERHEIDE, J., and JAHANIAN, F., "Internet inter-domain traffic," in *Proc. ACM SIGCOMM*, (New Delhi, India), Aug. 2010.

[70] LAFFONT, J.-J., MARCUS, S., REY, P., and TIROLE, J., "Internet interconnection and the off-net-cost principle," *The Rand Journal of Economics*, vol. 34, no. 2, 2003.

[71] LEBER, M., "IPv6 Internet broken, cogent/telia/hurricane not peering." http://www.merit.edu/mail.archives/nanog/msg01006.html, Oct. 2009.

[72] LEE, S., ZHANG, Z.-L., and NELAKUDITI, S., "Exploiting as hierarchy for scalable route selection in multi-homed stub networks," in *Proc. Internet Measurement Conference*, (Taormina, Italy), Oct. 2004.

[73] LEVEL 3 COMMUNICATIONS, "Level 3 statement concerning Comcast's actions." `http://www.level3.com/index.cfm?pageID=491&PR=962`. Retrieved: June 2011.

[74] MAS-COLELL, A., WHINSTON, M., GREEN, J., and DE CIČNCIES ECONŇMIQUES I EMPRESARIALS, U. P. F. F., *Microeconomic theory*, vol. 981. Oxford university press New York, 1995.

[75] "MaxMind GeoIP Country." `http://www.maxmind.com/app/geolitecountry`. Retrieved: June 2011.

[76] MCFADDEN, D., "Conditional logit analysis of qualitative choice behavior," *Frontiers Of Econometrics*, 1974.

[77] MCKENZIE, L. W., "On the existence of general equilibrium for a competitive economy," *Econometrica*, 1959.

[78] MILLER, G., "Overlay routing networks (Akarouting)." `http://www-math.mit.edu/~steng/18.996/lecture9.ps`, Apr. 2002.

[79] MO, J. and WALRAND, J., "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, 2000.

[80] MOTIWALA, M., ELMORE, M., FEAMSTER, N., and VEMPALA, S., "Path Splicing," in *Proc. ACM SIGCOMM*, (Seattle, WA), Aug. 2008.

[81] NABIPAY, P., ODLYZKO, A. M., and ZHANG, Z.-L., "Flat versus metered rates, bundling, and "bandwidth hogs"," in *6th Workshop on the Economics of Networks, Systems, and Computation*, (San Jose, CA), June 2011.

[82] "Cisco NetFlow." `http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html`. Retrieved: June 2011.

[83] "CS 8803: Next Generation Netowrks." `http://www.gtnoise.net/classes/cs8803/spring_2010/`, 2010.

[84] NORTON, W., "DrPeering.net." `http://drpeering.net`.

[85] NTT EAST, "Leased circuit price list." `http://www.ntt-east.co.jp/senyo_e/charge/digital.html`. URL retrieved January 2011.

[86] ODLYZKO, A. M., "Network neutrality, search neutrality, and the never-ending conflict between efficiency and fairness in markets," *Review of Network Economics*, vol. 8, pp. 40–60, Mar. 2009.

[87] OF OREGON, U., "RouteViews." `http://www.routeviews.org/`.

[88] ORE, "Wholesale leased lines price list." `http://www.otewholesale.gr/Portals/0/LEASED%20LINES_Pricelist_ENG_081110.pdf`. Retrieved: June 2011.

[89] PANG, J., AKELLA, A., SHAIKH, A., KRISHNAMURTHY, E., and SESHAN, S., "On the responsiveness of dns-based network control," in *Proc. Internet Measurement Conference*, (Taormina, Italy), Oct. 2004.

[90] "Peering Database." `http://www.peeringdb.com`.

[91] "PlanetLab." `http://www.planet-lab.org/`, 2010.

[92] POPESCU, A. and UNDERWOOD, T., "D(3) Peered: Just the Facts Maam. A Technical Review of Level (3)s Depeering of Cogent," in *NANOG 35*, Oct. 2005.

[93] "ProtoGENI." http://www.protogeni.net/, 2010.

[94] RAGHAVAN, B. and SNOEREN, A. C., "A system for authenticated policy-compliant routing," in *Proc. ACM SIGCOMM*, (Portland, OR), Aug. 2004.

[95] REKHTER, Y., LI, T., and HARES, S., *A Border Gateway Protocol 4 (BGP-4)*. Internet Engineering Task Force, Jan. 2006. RFC 4271.

[96] "Réseaux IP Européens Next Section Routing Information Service (RIS)." http://www.ripe.net/ris/.

[97] "RouteScience." Whitepaper available from http://www.routescience.com/technology/tec_whitepaper.html.

[98] SAVAGE, S., ANDERSON, T., and OTHERS, "Detour: A Case for Informed Internet Routing and Transport," *"IEEE Micro"*, vol. 19, pp. 50–59, Jan. 1999.

[99] SCHRAN, A., REXFORD, J., and FREEDMAN, M., "Namecast: A Reliable, Flexible, Scalable DNS Hosting System," *Princeton University, Technical Report TR-850-09*, 2009.

[100] SESHAN, S., STEMM, M., and KATZ, R., "A Network Measurement Architecture for Adaptive Applications," in *Proc. IEEE INFOCOM*, (Tel-Aviv, Israel), Mar. 2000.

[101] SHAKKOTTAI, S., SRIKANT, R., OZDAGLAR, A. E., and ACEMOGLU, D., "The price of simplicity," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 7, pp. 1269–1276, 2008.

[102] SPRING, N., PETERSON, L., BAVIER, A., and PAI, V., "Using planetlab for network research: myths, realities, and best practices," *SIGOPS Oper. Syst. Rev.*, vol. 40, Jan. 2006.

[103] SUBRAMANIAN, L., CAESAR, M., EE, C. T., HANDLEY, M., MAO, M., SHENKER, S., and STOICA, I., "HLP: A next generation inter-domain routing protocol," in *Proc. ACM SIGCOMM*, (Philadelphia, PA), Aug. 2005.

[104] TARIQ, M. B., ZEITOUN, A., VALANCIUS, V., FEAMSTER, N., and AMMAR, M., "Answering "What-if" Deployment and Configuration Questions with WISE," in *Proc. ACM SIGCOMM*, (Seattle, WA), Aug. 2008.

[105] TELEGEOGRAPHY, "Bandwidth pricing report." http://www.telegeography.com/product-info/pricingdb/download/bpr-2009-10.pdf. Retrieved: June 2011.

[106] UHLIG, S. and BONAVENTURE, O., "Designing bgp-based outbound traffic engineering techniques for stub ases," *ACM Computer Communications Review*, vol. 34, pp. 89–106, Oct. 2004.

[107] VALANCIUS, V. and FEAMSTER, N., "Multiplexing BGP sessions with BGP-Mux," in *Proc. CoNEXT*, Dec. 2007. Poster session.

[108] VALANCIUS, V., FEAMSTER, N., JOHARI, R., and VAZIRANI, V., "MINT: A Market for Internet Transit," in *ReArch - Re-Architecting the Internet*, Dec. 2008.

[109] VALANCIUS, V., FEAMSTER, N., REXFORD, J., and NAKAO, A., "Wide-Area Route Control for Distributed Services," in *Proc. USENIX Annual Technical Conference*, (Boston, MA), June 2010.

[110] VALANCIUS, V., LUMEZANU, C., FEAMSTER, N., JOHARI, R., and VAZIRANI, V., "How many tiers? Pricing in the internet transit market," in *Proc. ACM SIGCOMM*, (Toronto, Ontario, Canada), Aug. 2011.

[111] "VINI: Virtual Network Infrastructure." `http://www.vini-veritas.net/`.

[112] WANG, H., XIE, H., QIU, L., SILBERSCHATZ, A., and YANG, Y., "Optimal isp subscription for internet multihoming: Algorithm design and implication analysis," in *Proc. IEEE INFOCOM*, (Miami, FL), Mar. 2005.

[113] WEISS, A., "Computing in the clouds," *netWorker*, vol. 11, no. 4, pp. 16–25, 2007.

[114] WENDELL, P., JIANG, J., REXFORD, J., and FREEDMAN, M., "DONAR: Decentralized server selection for cloud services," in *Proc. ACM SIGCOMM*, (New Delhi, India), Aug. 2010.

[115] XU, W. and REXFORD, J., "MIRO: Multi-path Interdomain ROuting," in *Proc. ACM SIGCOMM*, (Pisa, Italy), Aug. 2006.

[116] YANG, X., WETHERALL, D., and ANDERSON, T., "Source selectable path diversity via routing deflections," in *Proc. ACM SIGCOMM*, (Pisa, Italy), Aug. 2006.

[117] ZHANG, Z., ZHANG, M., GREENBERG, A., HU, Y. C., MAHAJAN, R., and CHRISTIAN, B., "Optimizing Cost and Performance in Online Service Provider Networks," in *Proc. 7th USENIX NSDI*, (San Jose, CA), Apr. 2010.