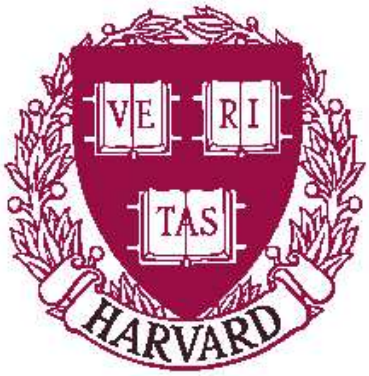
Perspective Access Networks

Perspective Access Networks

Geoffrey Lewis Goodell

TR-12-06



Computer Science Group
Harvard University
Cambridge, Massachusetts

Perspective Access Networks

A dissertation presented
by

Geoffrey Lewis Goodell

to
The Division of Engineering and Applied Sciences
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the subject of

Computer Science

Harvard University
Cambridge, Massachusetts
July 2006

to my mother, Elaine

Thesis advisor

Mema Roussopoulos

Author

Geoffrey Lewis Goodell

Perspective Access Networks

Abstract

Perspective Access Networks provide an infrastructure from which users can specify the location from which they wish to view the Internet. The ability to specify location has become necessary as the Internet has become increasingly inconsistent. An increasing preponderance of middleboxes, location-dependent services, and large-scale content filtering have contributed to this situation.

Our work offers the following contributions. First, we propose an infrastructure that routes traffic to a location from which a given resource can be viewed, taking instructions from user-specified attributes describing the desired location. Second, we analyze the tradeoff between the expressivity of user requests and the finite resources available within the network for propagating metadata about available perspectives. Third, we stipulate a set of real scenarios that fall within the limits of what can reasonably be handled by a system appropriately tuned to manage the tradeoff, and we argue that the specific algorithm we propose can handle the scenarios effectively.

Contents

List of Figures

List of Tables

SOMETHING there is that doesn't love a wall,
That sends the frozen-ground-swell under it,
And spills the upper boulders in the sun;
And makes gaps even two can pass abreast.
The work of hunters is another thing:
I have come after them and made repair
Where they have left not one stone on a stone,
But they would have the rabbit out of hiding,
To please the yelping dogs. The gaps I mean,
No one has seen them made or heard them made,
But at spring mending-time we find them there.
I let my neighbour know beyond the hill;
And on a day we meet to walk the line
And set the wall between us once again.
We keep the wall between us as we go.
To each the boulders that have fallen to each.
And some are loaves and some so nearly balls
We have to use a spell to make them balance:
"Stay where you are until our backs are turned!"
We wear our fingers rough with handling them.
Oh, just another kind of out-door game,
One on a side. It comes to little more:
There where it is we do not need the wall:
He is all pine and I am apple orchard.
My apple trees will never get across
And eat the cones under his pines, I tell him.
He only says, "Good fences make good neighbours."
Spring is the mischief in me, and I wonder
If I could put a notion in his head:
"Why do they make good neighbours? Isn't it
Where there are cows? But here there are no cows.
Before I built a wall I'd ask to know
What I was walling in or walling out,
And to whom I was like to give offence.
Something there is that doesn't love a wall,
That wants it down." I could say "Elves" to him,
But it's not elves exactly, and I'd rather
He said it for himself. I see him there
Bringing a stone grasped firmly by the top
In each hand, like an old-stone savage armed.
He moves in darkness as it seems to me,
Not of woods only and the shade of trees.
He will not go behind his fathers saying,
And he likes having thought of it so well
He says again, "Good fences make good neighbours."

— Robert Frost, *Mending Wall*

Acknowledgments

This dissertation would not have been successful without the guidance and support of a large number of individuals. Since it is not possible to list them all here, the following partial listing contains many grievous omissions:

Mema Roussopoulos, my thesis advisor, who believed in me since the initial stages of my thesis work. For several years, she provided insightful commentary during our regular meetings, and she was consistently supportive of my proposed research directions. I am honored to be her first graduate student.

Scott Bradner, technical consultant and Internet guru extraordinare, who guided me through a maze of Internet standards, government regulations, news stories, and technical issues. During our numerous lunchtime meetings, he taught me how to present my ideas clearly and effectively. His devotion to my project persisted even when mine wavered.

David Parkes, whose brilliant insights led me to strongly consider the social and economic impact of this project. He always managed to find the right questions to ask about a perspective-based approach to organizing the Internet, and he never hesitated to challenge my assumptions.

H. T. Kung, who coined the term "Perspective Access Network" to describe my academic contribution. He has been instrumental in directing the focus of my research and isolating the most interesting problems.

Susan Wieczorek, graduate program administrator in my department. As one previous student observed, she "made being a graduate student tractable." Over the past five years, she spent many hours listening to my concerns, working with me to navigate the bureaucracy, and assisting me with my most important decisions.

Paul Syverson of the Naval Ressearch Laboratory, whose incredibly clear thinking helped me distill my intuitions about network-based authentication into coherent arguments. He worked with me on a short paper that became the basis for Section 6.2.

Roger Dingledine and Nick Mathewson of the Free Haven Project. They are the principal authors of *Tor*, the popular and well-documented anonymity network that served as a substrate for the implementation of my project. They spent many hours working with me to determine the right abstractions for the interface between my Perspective Access Network tool and their onion routing system.

Chris Palmer and Lars Kellogg-Stedman, who manage the computing infrastructure for my division at Harvard. They supported my research despite significant legal and political controversy, and their commitment to providing an environment for my research transcended the call of duty.

Members of the Berkman Center for Internet and Society, particularly Phil Malone, Jonathan Zittrain, Charles Nesson, and John Palfrey, who provided an essential connection to Harvard Law School. They guided my understanding of the legal landscape and the potential uses of my research in civil rights applications.

Viktor Mayer-Schönberger of the Kennedy School, who guided my understanding of incentives within the regulatory environment.

Rachel Greenstadt, who provided a sounding board for my most bizarre ideas and ensured that I did not forget to eat or exercise.

Max Van Kleek and Elizabeth Stark, who ensured that I did not lose touch with the latest developments in electronic music.

Staff members Marilyn O'Brien and Tristen Hubbard, who ensured that the flow of information, money, and equipment always ran smoothly and efficiently.

My officemates, Chen-Mou Cheng, Pai-Hsiang Hsiao, and Dario Vlah, who provided a fantastic work environment, offered technical consultation, tolerated my loud telephone conversations, and rebooted my computers.

My colleagues at AT&T Laboratories, Patrick McDaniel, Avi Rubin, John Ioannidis, Timothy Griffin, and William Aiello, who offered me an opportunity to explore interdomain routing in depth. My work at AT&T taught me about inconsistency within the Internet, and in this sense my dissertation is a continuation of that work.

My colleagues at Goldman, Sachs & Co., Roberto Caccia, Ryan McCorvie, Sven Khatri, James Sarvis, and Elisha Wiesel, who convinced me that I would be able to finish my dissertation, even when I had substantial doubt.

My flatmate, Sebastien Lahaie, who convinced me that it was in fact possible to beat Pedro Sander at poker, even when I had substantial doubt.

Hillary Hurst, who believed in me and supported me during my darkest hour.

Carla Pellicano, an indispensable confidante throughout the past decade, celebrations and tribulations included.

Elaine Goodell, who taught me to always demand solutions rather than problems. Despite incessant responsibilities of her own, she listened to all of my problems anyway, and she never failed to offer love and support.

Finally, I would like to extend gratitude to my family members, friends, and colleagues, who supported my research during the past five years. Their devotion and confidence provided the instrumental motivation that I needed to continue, and I am forever in their debt.

Chapter 1

Introduction

The Internet is not flat: the set of resources that a user can access via the Internet is determined in part by how that user is connected. The network itself acts to limit access to particular resources in such a manner that access to certain resources is restricted to those observing them from particular locations.

Network location can be used to restrict or modify access to resources in two ways. First, network elements such as routers, network address translators, and firewalls may filter, modify, redirect, or naturally limit traffic based upon the location indicated by the network address of a given source or destination. Second, a server may choose to selectively refuse service or provide different service based upon the network location from which the traffic is apparently originating. In both cases, by providing a client the ability to specify where it wants to appear in the network, we can mitigate the network access constraints imposed by its particular attachment point. Of course, creating a tool for this purpose creates a point of contention, since clients could potentially establish end-to-end connections with other parties in defiance of

policies introduced by intervening network carriers.

Clark et al. characterize the ongoing arguments concerning Internet governance as a *tussle* in which various parties seek to manipulate the various intrinsic technical mechanisms of the Internet to their advantage. For example, governments, corporations, or network access providers might deploy firewalls, and end users might establish tunnels to circumvent them (28). Designers of network elements and services need to know what information they can use as a basis for authentication and access control. Indeed, the current political climate includes substantial discussion of how to determine the right way forward, and there is a clear need for well-defined architectural boundaries.

Our work addresses the tussle by providing a mechanism that allows Internet users to overcome location-based limitations. Many Internet services use network location as an intrinsic basis for determining what resource to provide, and the network itself often chooses the set of resources that are available to clients. As a result, providing location-independent access to resources is sometimes inherently impossible. Furthermore, it is not possible for a user to know whether she is viewing a resource from a "neutral" perspective or not. The only way to provide consistent access to resources is to allow clients the ability to send and receive Internet traffic from the specific locations that provide access to the desired resource. We propose a *Perspective Access Network* (PAN), an overlay network that allows users to specify not only the resources that they want to view but the *perspectives* from which they want to view them. Perspective Access Networks provide a clear boundary in which parties on opposite sides of the tussle can use a consistent interface to make their own policy decisions. By

separating the argument about policy from the argument about architecture, we hope to facilitate the development of policies that more appropriately address the needs of parties with conflicting interests.

Later in the same discussion, Clark et al. suggest that in an ideal world, customers would be able to use a paradigm akin to source routing to select the paths (and, implicitly, the network carriers) that their packets take en route to particular destinations. The authors argue that overlay networks could provide a useful tool for customers by allowing them to avoid undesirable paths imposed by their providers. Since providing discriminatory access to resources is often in the best interests of providers (137), having a tool that allows circumvention of undesirable routes may be in the best interests of customers. PANs provide an architecture that can potentially allow these customers to avoid undesirable access restrictions imposed by their providers. Customers may use PANs not only to avoid suboptimal paths to their desired destinations but also to access destinations that they cannot access directly as the result of mechanisms imposed by the network. Providers could respond by restricting access to Perspective Access Networks, although such a response could ultimately lead to more overt conflict between providers and their customers. Indeed, we view Perspective Access Networks largely as counterbalance against fragmentation that could happen in the future; the existence of technical ways of overcoming fragmentation might make fragmentation less attractive as a method for regulating the behavior of Internet users.

Naturally, since network location is often used for purposes of identification and authorization, the ability for clients to mask their actual point of attachment as they

connect to Internet services raises important concerns about trust, identity, abuse, authentication, and incentives to deployment. We will address these points in later chapters.

1.1 Motivation

Recently, new threats to Internet consistency have received media attention. The issues fall into two categories: conflict concerning *naming* and the use of *geolocation* to restrict access to resources. First, a number of nations have raised formal objections to oversight of ICANN by the United States, and a number of private organizations such as UnifiedRoot have emerged to offer alternative namespaces (111). Global agreement on Internet governance is becoming increasingly difficult (146) which means the potential for inconsistency in naming resulting from multiple DNS roots or addresses that are not globally unique will only increase. To a significant extent, the Internet depends upon everyone having access to the same set of names. The threats, therefore, are as follows: (a) the same name does not exist in both of two locations (lack of global consistency), and (b) the same name refers to different resources in different locations (lack of global uniqueness).

Second, a perceived increase in online criminal activity has created viable business models for businesses that provide geolocation services marketed for their benefits in fraud resolution and digital rights management[1]. For example, a number of companies use these geolocation services to obtain information about how a user is connected

[1]CyberSource, `http://www.cybersource.com/`; NatGeo `http://www.natgeo.com/`; Quova, `http://www.quova.com/`

to the Internet (such as IP address and ISP data) to determine whether the user is likely to be fraudulent. This has caused a number of legitimate online transactions to be denied when users are not connected at their usual point of attachment (80). Finally, various governments and service providers around the world have deployed network technology that (accidentally or intentionally) restricts access to certain Internet content (100; 48).

Combined with the various well-known sources of fragmentation that we will describe in detail later in this section, these new concerns provide ample motivation for development of a technique that affords users the ability to specify not only the network location of Internet resources they want to view but also the *perspectives* from which they want to view them. In this thesis, we present the design, implementation, and evaluation of a *Perspective Access Network*, an overlay infrastructure for sharing perspectives. Our prototype, called *Blossom*, consists of an unstructured, peer-to-peer overlay of *forwarders* carrying TCP traffic that act as intermediaries between nodes that cannot communicate directly.

1.1.1 Fragmentation Defined

We use the term *fragmentation* to refer to the manner in which access to Internet resources is inconsistent with respect to network location.

There are many causes of fragmentation, ranging from accidental (routing failures, misconfigured policies, unreliable network elements) to deliberate (content filtering, network address translation, firewalls, malicious service providers). We are interested primarily in:

- (a) purposeful, data-specific or content-specific fragmentation resulting from middleboxes that take action, such as filtering or redirection, based upon the traffic it encounters,
- (b) routing policies implemented by particular network access providers, including policies derived from limitations in business and trust relationships between BGP peers, and
- (c) DNS names that are not globally available or not globally unique, perhaps as a result of political disputes over the role of ICANN, the organization responsible for provisioning Internet names and addresses.[2]

Above all, we believe that fragmentation is inevitable: the address isolation afforded by NAT devices is commercially precious, and global agreement on Internet governance will only become increasingly difficult as the number of participants grows.

Various aspects of network design contribute to fragmentation. First, prevailing Internet architecture allows for the existence of points of control within the network. Specifically, it is possible to leverage network infrastructure such as routers and switches to manage access to resources; policy is often explicitly determined by mechanisms applied at the network layer. Second, despite attempts to regulate and provision Internet activity, the core of the Internet is non-hierarchical. In particular, the lack of central authority allows for the possibility that regions of the network under different management may have conflicting interests; the result is that while providing access to a particular resource may be of interest to one network, it may not be of interest to another.

[2]Internet Corporation for Assigned Names and Numbers, `http://www.icann.org/`

Our view is that fragmentation itself is not intrinsically desirable or undesirable. Instead, it is a naturally occurring consequent of the decentralized nature of large networks. As the Internet continues to grow in size, scope, and significance to world-wide economic activity, it seems natural that disputes among local authorities and service providers will become more contentious.

The Internet is neither a perfect hierarchy nor a uniform set of equal partners (97). Reality lies somewhere in the middle: each Internet service provider manages some number of *autonomous systems* (ASes), and autonomous systems are arranged into general *tiers*, such that providers within each successive tier tend to offer service to *customers* who exist within the next tier. The result is that a small handful of providers form a loosely-defined "core" of the Internet. The term *default-free zone* describes the set of autonomous systems within this "core" who do not use default addressing to identify an upstream service provider. Instead, such autonomous systems use specific interfaces for specific ranges of destination addresses (prefixes) without systematically assigning some substantial proportion of traffic as unclassified, ready for delegation to some other autonomous system. While the Internet today is arranged such that the autonomous systems that are part of the default-free zone form a single (if hard to define) cluster, this arrangement will not necessarily always be the case.

For example, Microsoft and Nokia applied for an additional top-level DNS domain for use by wireless devices.[3] Similarly, China has proposed additional DNS roots, managed by servers under the control of the Chinese government. There is also some speculation that China might introduce its own address space and separate

[3]Mobile Data News, `http://www.mda-mobiledata.org/mda/documents/MDNAPR04.pdf`

default-free zone. If implemented, such a network would effectively become a second Internet, connected in various places to the "core" Internet, but existing independently. Reasons for establishing sovereignty might include (a) the ability to wholly manage allocation of names, addresses, or routing infrastructure.

In both cases, there exist economic or socio-political arguments for why an organization may want to use a walled-garden strategy of separation from the main Internet "core" in order to capture control. It is, therefore, useful to consider a system for ensuring universal access to resources in an Internet divided in such a manner.

Throughout this discussion, we refer to *names* and *identifiers*. We use the term *identifier* to refer to a symbol that establishes the identity of a particular entity in a particular context, and we use the term *name* to refer to a sequence (possibly one) of identifiers used to refer to a particular entity. DNS names as conceived in the present Internet may be *fully-qualified*, identifying an entity with respect to a categorically acknowledged root. All names are fully qualified within any given perspective. However, if a Perspective Access Network spans two environments that do not share a common root, then there is no sense by which a single name can be considered fully qualified throughout the network.

1.1.2 Causes of Fragmentation

Next, we characterize several of the various ways in which fragmentation occurs. While our system should be capable of addressing fragmentation generally, it is more well-suited to certain kinds of fragmentation than to others. Table 1.1 provides an overview of the various forms of fragmentation and specifies those that we address.

Type	Addressed by PANs?
Interdomain Routing Policy	Yes
Interdomain Routing Misconfiguration	Partially
Interdomain Routing Instability	No
Firewalls	Yes*
Network Address Translation	Yes
Content Filtering	Yes*
Explicit Address Filtering	Yes*
Transparent Proxies and Caches	Yes
Anycast	Yes
DNS Manipulation	Yes

Table 1.1: Causes of Fragmentation. *(*To regulate Internet use, network access providers may block access to Perspective Access Networks.)*

- Interdomain Routing Policy. Each autonomous system that uses BGP for interdomain routing is responsible for establishing its own policy specifying from which peers to accept particular advertisements and to which peers to send particular advertisements. Generally such policy is dictated by independent decisions on the part of individual Internet service providers. However, it is important to recognize that policies restrict the advertisement of routes in general, and there are no guarantees that routes to all prefixes will be received by all autonomous systems. Nearly all agreements between providers to exchange BGP routes fall into one of two categories (77):

  - The *customer relationship*, in which a provider advertises to a customer (a) its internal routes, (b) routes from all other peers, and (c) routes from its other customers.
  - The *peering relationship*, in which each peer advertises to a peer (a) its internal routes and (b) routes to its customers.

Thus, the *customer relationship* consists of a provider offering either a full routing table or a default route to its customer, and the *peering relationship* consists of a link between two directly connected providers arranged such that their respective customers can communicate via that link. Internet service providers may not engage in the right set of relationships with the right set of peers and providers to obtain access to all networks throughout the Internet.

- Interdomain Routing Misconfiguration (Accidental). Misconfiguration of routers that participate in the BGP protocol is a significant cause of observed routing failures. A routing failure occurs when a BGP speaker advertises something that should not be advertised or suppresses something that should be advertised. Mahajan et al. organize classes of BGP misconfiguration into two main categories: *origin misconfiguration*, which consists of the advertisement of a prefix that a BGP speaker is not authorized to advertise, and *export misconfiguration*, which consists of the advertisement of a route in a manner inconsistent with the policy of the exporter (82; 59). Accidental misconfiguration may result from simple data entry errors, or it may result from misunderstanding about the implications of certain BGP policy decisions. Accidental misconfiguration has resulted in unreachability and suboptimal routes in a few cases.

- Interdomain Routing Misconfiguration (Purposeful). To our knowledge, accidental misconfiguration accounts for most large-scale failures of BGP routing, but the potential for malicious misconfiguration exists as well. Both external hackers and malicious employees could potentially introduce rout-

ing policies inconsistent with the policy of the ISP. While interdomain routing security is a serious problem with a number of proposed solutions (74; 54), malicious advertisements have so far not substantially posed an active threat to the infrastructure. We describe some of the specific vulnerabilities in Section 5.2.3.

Indeed, both accidental and purposeful configuration of policy can lead to fragmentation. For reasons of trust, quality of service, sorting priorities, or political reasons, providers may or may not opt to accept, advertise, or use routes offered by their neighbors. The result is that not all networks actually have access to all other networks, even if all have Internet connectivity. Additionally, providers tend not to have agreements about filters in general; inconsistent filtering may result in incomplete access to available Internet resources. Providers may fail to provide perfect connectivity because they do not consider all of the ramifications of their policy choices, or because the process of verifying that policy choices provide full connectivity is prohibitively difficult or expensive. Indeed, border routers sometimes contain up to hundreds or even thousands of lines of BGP policy statements. However, providers sometimes fail to provide full connectivity with full knowledge of the decision as well; such providers may know that a particular network will be unreachable to their customers but decide that the costs of providing that connectivity outweigh the inconvenience to their customers.

Perspective Access Networks do not entirely resolve interdomain routing misconfiguration concerns (accidental or purposeful), since they are susceptible to

similar misconfiguration. Nevertheless, the ability to view the Internet from different perspectives may provide a useful resource for debugging purposes.

- Interdomain Routing Instability. Researchers have observed that interactions between BGP policies often lead to persistent interdomain routing oscillation (138; 56), and that interdomain routing oscillation in turn leads to degraded network performance (59). The problem is endogenous to the nature of the policies themselves and the way in which routes are propagated through the network (58). Methods such as route-flap dampening are commonly used to mitigate the adverse effects of policy-driven oscillation, but such methods are inherently imperfect and can themselves contribute to extended periods of network fragmentation. We briefly describe the causes of BGP routing instability in Section 5.2.3.

  Perspective Access Networks do not represent an attempt to mitigate interdomain routing instability among autonomous systems, nor do they intend to address transient network failures. Resilient Overlay Networks (RON) serve this purpose; refer to Chapter 2 for more information.

- Firewalls. A *firewall* is a device for filtering traffic according to a set of rules. Typically the rules specify a set of patterns such that IP packets with transport-layer headers (e.g., TCP, UDP, ICMP) that match one or more of the specified patterns (e.g., a prefix that is a known source of spam) are simply dropped. Most commonly, firewalls are used to block inbound requests for services (e.g., filter inbound traffic with the TCP SYN flag set) or particular services themselves (e.g., filter based upon TCP or UDP port number). There are many

well-known methods for circumventing firewalls, including establishing tunnels using allowed protocols, binding services to nonstandard ports, or setting up additional gateways.

Firewalls are sometimes used to enforce legal policy. For example, various regulations such as the Sarbanes-Oxley Act (133) and the Gramm-Leach-Billey Act (132) have encouraged accounting departments and financial institutions seeking legal compliance to (a) maintain archives of traffic traversing the borders of their networks or (b) generally limit all traffic except for a small set of services. More commonly, however, firewalls are deployed for security reasons.

Since most remote attacks consist of random, automated probing for services with particular vulnerabilities over an address range, systematically blocking these probes at the network layer serves as a practical, though imperfect, line of defense. However, this comes at the cost of effectively enacting a policy, even when such policy is not required by organizational goals. Many organizations that use firewalls misunderstand the threat models; studies have shown that up to seventy percent of hacking activity within corporate environments are knowingly instigated or facilitated by insiders (106). Also, there is a well-established premise that in many environments, firewalls stifle innovation by disallowing the deployment of services not explicitly sanctioned in advance. PANs can be used to address fragmentation resulting from firewalls.

- Network Address Translation. Suppose that a network administrator wants to connect one existing network to another without requiring that the individual networks have mutual knowledge of the addresses contained within

the other network.[4] In this case, there are two separate address spaces arranged such that peers in one address space cannot communicate directly with peers in the other address spaces. *Network address translators* are essentially routers with interfaces in both address spaces; they systematically rewrite headers of packets to allow communication between both address spaces, maintaining state for individual connections, usually with a table that maps individual TCP connections to port numbers.

One might think that widespread deployment of NAT devices is primarily the result of fears that available address space is insufficient, but Classless Inter-Domain Routing (50) allows sufficient flexibility that a suitably large organization can obtain sufficiently many addresses to satisfy its needs quite inexpensively. Indeed, organizations often deploy NAT devices or enable NAT functions in firewalls for the same "security reasons" used to justify the deployment of the firewalls themselves. NAT devices are even more pernicious than some firewalls in the sense that NAT devices *must* store connection state, violating the end-to-end principle (114), making systems at the ends of the network dependent upon the reliability of systems in the interior of the network. Also, because resources to identify individual connections are finite, NAT devices typically recycle entries in their tables. Since NAT devices have no way of knowing whether a particular conversation is still active or whether both hosts have abandoned the connection, the process of recycling entries often leads to disconnected sessions.

[4]The most common example is an individual or organization that reserves one IPv4 address from an upstream Internet service provider and intends to use this single address to connect an entire network to the Internet.

PANs can be used to address fragmentation resulting from NAT.

- Content Filtering. Some firewalls are configured to filter packets based upon application-layer content; this technique can be used for large-scale censorship of sensitive content. For example, British Telecom recently deployed a system to restrict access to certain web pages (16); one way of implementing this restriction is to filter HTTP packets based upon the URL specified in the request. Certainly this approach is not immune to false positives (116; 85). Regardless, however, the scale at which governments and providers are considering deployment of such technologies indicates the potential for misuse. Many large corporations purposefully use some form of content filtering to limit the use of company-administered machines to access certain kinds of Internet content.

  A substantial number of governments around the world use filtering to restrict access to Internet resources on the basis of the content that their citizens might be able to access. For example, regimes have been known to filter access to news stories, political discussion, pornography, hate speech, religious speech, and other categories of content. Different regimes have different policies that involve filtering different content categories, and the technology used determines the extent to which these regimes are successful. The Open Net Initiative (ONI) periodically publishes a series of reports describing these policies and cataloguing the extent to which filtering actually occurs around the world (100).

  One of the most important uses of Perspective Access Networks is the circumvention of content filtering; we discuss the technique in greater detail in Chapter

5.

- Explicit Address Filtering. Providers of Internet service may also configure their firewalls to explicitly block packets based upon their source or destination addresses. Technically, the implementation of such filtering rules may be no different from ordinary firewall rules specified by local networks at the edges of the Internet. However, deploying such filtering technology in the middle of the network suggests greater distance between those subject to the policy and those enacting it. Consider the Pennsylvania state statute (no longer in effect) that specified that traffic to and from certain hosts must be filtered by service providers (84). It may have been nontrivial for providers to deploy government-mandated filtering on a large scale, but the result was an infrastructure for systematically preventing access to Internet resources. The existence of NAT, hosts running more than one service, and web servers hosting more than one page suggest that this method would have been even more prone to false positives than filtering based upon application-layer content. PANs can be used to circumvent filtering of this sort.

- Transparent Proxies and Caches. Transparent proxies are routers configured to either (a) redirect, or (b) intercept and forward, requests for a particular resource (e.g., an HTTP request for a web page) to a proxy. The proxy subsequently issues the request on behalf of the original sender, receives the response, and then forwards the response to the original sender. Often, transparent proxies also cache replies: in cases when many users of a local network request the same web page, such caching can sometimes decrease the load on

an Internet uplink by allowing the page to be requested only once, serving the same cached copy to all requesters. Recently, the IETF proposed a natural extension of this technique called Open Pluggable Edge Services (OPES), which allows intermediaries to customize a data stream as part of a service (7). For example, OPES could be used to transparently insert advertisements specific to particular networks or geographies into Web content.

The technique of transparently mutating Internet traffic has several complications, however. For example, a client whose request is intercepted by a transparent proxy may fail to issue cache-control directives that could be used to specify that the source of the request does not want the copy (1) to be cached. More generally, such systems pose a threat to network transparency by injecting intelligence into the center of the network (127; 10). PANs can be used for circumvention of proxies and caches.

- ANYCAST. Sometimes a single service is provided by multiple hosts, and when a user or application wants to locate the service, it does not matter which of these hosts actually provides the service. The term *anycast* refers to a system that allows the user or application to specify the service without requiring a response from any one server in particular: the network is responsible for determining which host actually provides the service (103; 2). By using anycast to determine which DNS server receives a particular query, some service providers are able to usefully balance load across multiple servers or even allow the requester of the service to find a server topologically proximate to itself. While there are clear efficiency benefits, we observe that the requester is unable to specify the server

specifically—the network must choose on behalf of the requester. A requester might want to access a specific server within an anycast group. Today, anycast is used primarily for DNS servers (75). PANs may conceivably be used to allow clients to request a specific member of an anycast group.

- DNS Manipulation. The Domain Name Service determines to which IP addresses a host sends a request for a resource identified by a particular hostname. DNS can be used to provide different IP addresses for the same hostname based upon the location of the requester in the network. For example, Google has distributed servers throughout the world to handle requests, and the popular search service uses DNS to direct peers to particular servers in the network based upon the location of the peers in the network. On 26 July 2004, an insidious worm launched a multitude of messages at Google servers, but since not all servers received the same number of requests, some servers were effectively disabled while others remained functional (113). Since DNS continued to direct some peers to nonfunctional servers, some users were unable to access the resource. If there had existed a way of specifying which server to use, then it may have been possible for such users to access the unaffected servers. If we presume that each PAN forwarder uses some particular domain name server, then PANs may be used to indirectly select which server to use.

1.2 End-To-End System Design

At the core of our argument lies a sense that the fragmentation problem is the result of the network itself interfering with communication between hosts. For decades, researchers have argued in favor of the normative notion that the network ought to be a transparent medium, providing connectivity but not interfering with its traffic. The *end-to-end principle* states that the costs of providing special functionality at low levels of a system generally outweigh the benefits. As originally described, the end-to-end principle refers to special network functions, including delivery guarantees, secure transmission of data, duplicate message suppression, guaranteeing FIFO message delivery, and transaction management (114).

Implementing these functions at the network layer leads to a design that is (a) less flexible, since the parties with an interest in using the functions have no control over how they are implemented; (b) more intrusive, since parties without an interest in using the functions are subjected to their provisions; (c) more brittle, since additional functionality in the network means additional opportunity for failure; and (d) more cumbersome, since upgrading entire infrastructures so that a few nodes on the edges can take advantage of a new protocol feature might be prohibitively expensive. Furthermore, technical solutions that violate end-to-end principles may stifle innovation by unnecessarily constraining the set of assumptions that devices on the edges of the network can make about how the network will behave (51; 11).

As technology became more sophisticated, networking experts extended the argument to encompass higher layers in the protocol stack and issues involving connection

state, authentication, and host mobility (72). As the set of Internet users continued to grow, the end-to-end argument also expanded to serve as a rallying point for *network neutrality*, the general principle that the networks of which the Internet is composed should not impose restrictions on the traffic that they carry. So, the argument goes, the network ought to be *transparent*: a collection of neutral pipes that promiscuously convey all traffic from potentially any source to potentially any destination.

Indeed, conflicts of interest have emerged along with the "tussles" described earlier (28). In particular, as the Internet population expanded, the interests of its users began to diverge. Security became an issue, and implicitly trusting the set of all users was no longer practical. Technology that differentiated hosts based upon their network location demonstrated a certain effectiveness in mitigating attacks, despite the fact that such technology (a) violated network transparency by interposing between interlocutors (for example, in the case of transparent proxies), and (b) violated important abstractions (for example, the idea that the Internet is globally consistent) by intrinsically binding policy to low-level mechanisms. The emergence of the Virtual Private Network (VPN) is testament to the crudeness of approaches that rely upon network isolation. Trust boundaries within organizations often do not map directly to underlying network topologies, either because the relevant sets of people change frequently or because the relevant sets of people are not physically collocated. As a result, some organizations adopted a "trust envelope" paradigm, in which some combination of network boundaries and special-purpose authentication are used to determine whether a given user is permitted to access a particular resource. Some of these organizations have deployed increasingly complex infrastructure to extend their

trust envelopes in arbitrary ways (27).

Perspective Access Networks allow the extension of the trust envelope in a general way that is expressive, uniform, and architecturally stable; this is accomplished by adding authentication as described in Section 3.5.

1.3 Addressing Fragmentation

We assert that the amorphous nature of the Internet facilitates its growth and that fragmentation is part of this amorphous nature. Hence, our approach seeks to harness the benefits of fragmentation rather than stifle fragmentation entirely. We design Perspective Access Networks around four central objectives: locality, access through obstructions, decentralized resource allocation, and deployability. We describe each of these goals in detail in Section 1.4, and we demonstrate that one infrastructure can be used to provide all of these advantages.

This thesis characterizes ways in which systemic fragmentation occurs today, examines why existing architectures fail to avoid fragmentation, and considers the design of existing systems created to mitigate aspects of the phenomenon. Our analysis provides a better understanding of what characteristics are required by a general-purpose system for facilitating communication in a fragmented network, which in turn allows us to argue in favor of particular design choices. We provide a proof-of-concept implementation and provide design guidelines for other implementations. We conclude with an exploration of the impact and benefits of such a system.

Note that our system design achieves its seemingly conflicting goals of locality and access through obstructions without depending upon universal naming. Indeed,

one of the key features of the Internet today is that names used to identify resources are universal: they depend only upon the resource and are not defined by who is requesting the resource. We argue that universal naming is not indispensable, and we believe that by relaxing this constraint we can achieve a considerably more flexible network.

The architecture that we present does not require all Internet users to have the same notion of what region of the Internet constitutes the "core" or which set of real-world organizations are responsible for Internet governance. Perspective Access Networks allow us to consider a heterogeneous Internet whose management reflects the management of the real world rather than imposing organizational structure where hierarchy need not exist.

Previous work on overcoming network fragmentation to facilitate end-to-end connectivity requires extensive changes to operating systems (such as deployment of new protocol stacks), requires the explicit participation of ISPs and content providers, or imposes a global hierarchical organization of the Internet. We relax these constraints to provide *ease of deployment* and have built a system we have deployed on the Tor anonymity network (38) and on PlanetLab (66). Our approach does not require changes to the operating system or protocol stack, does not require active participation of ISPs, and does not require special configuration of in-band network-layer elements such as routers or middleboxes.

1.3.1 Primary Contributions

We use a multi-faceted approach to construct an argument for the relevance and usefulness of Perspective Access Networks:

- Solution Space. We consider the set of existing solutions offered by the networking and systems communities to address problems similar to those that we address. We observe that the existing approaches fall into four main categories. Some systems, including RON (5), concern themselves primarily with providing improved robustness against transient network failure. Other systems, including Mobile IP (104) and Unmanaged Internet Protocol (46), seek to provide *mobility*, allowing end hosts to move around throughout the network without losing their ability to communicate with peers. A third class of systems, including TRIAD (23) and DOA (142), assert the inevitability of middleboxes and seek to provide a network architecture that allows such middleboxes to operate in accordance with the fundamental tenets of the original Internet design objectives. A fourth class of systems, including Platypus (121), seek to provide a means by which Internet service providers can define or negotiate a richer, more effective set of filtering policies.

- Architectural Objectives. We define a set of central architectural objectives for an infrastructure capable of routing traffic to an appropriate location for viewing a resource. At a high level, we are concerned with how to propagate metadata about perspectives through the network in a scalable way. In particular, to provide to clients the benefits of locality and access through middleboxes, we propose a language sufficiently expressive to describe the perspectives from

which they want to view the network and a process by which those perspectives can be reached. We also provide a policy framework for configuring PAN elements so that the policy needs of all players involved in the tussle are adequately satisfied.

- Implementation. We present *Blossom*, our realized prototype implementation of a Perspective Access Network. We outline a set of desiderata for a source-routing infrastructure that can be used as a transport layer for PAN, and we show how PAN can take advantage of such infrastructures.

- Analysis. We carefully examine Perspective Access Networks to assess the scalability, deployability, and usefulness of our design. We consider system scalability from both technical and policy standpoints, and we argue that our policy framework can be used to assure that incentives of those deploying PAN infrastructure are adequately satisfied. We provide a method by which the PAN directory infrastructure can manage the tradeoff between client performance and volume of routing information within the control plane, and we argue that there are important and significant uses of PANs that exist within the limits of what that method can handle. To inform our discussion and theoretical analysis, we provide quantitative results from a series of experiments that evaluate the behavior of PANs in a real-world environment.

- Incentives for Deployment. We consider a set of real-world scenarios that fall within the limits of what our system can handle. For example, users in China may seek unfiltered access to BBC News. Users in one branch office of an

enterprise my seek perspectives within an internal network segment of another branch office. Users may have an interest in geographically customized search results or the ability to view the Internet as seen from home while travelling. Network administrators may want to perform security audits from afar. Researchers, government agencies, and political organizations may want a means of quantifying political filtering. (Section presents a detailed discussion of these scenarios.)

1.3.2 A "Coreless" Internet

PAN allows us to study what the world would be like with a *"coreless"* Internet, i.e., an Internet without globally assigned names or addresses. A client using the PAN overlay can access a remote resource, provided that it can build a tunnel through the network, across fragments, to a remote forwarder that can access that remote resource. Like popular peer-to-peer filesharing networks, PAN allows end users to participate directly, but PAN users are sharing their *perspectives* rather than their content.

PAN also does not depend on global hierarchical organization of the Internet. Currently, both the addresses and the names used to identify resources on the Internet are allocated by a collection of governance organizations that are arranged hierarchically with a single organization at the top having overall "control." Our approach allows for an Internet without hierarchically ordained names and address spaces—that is, an Internet consisting of (possibly overlapping) network fragments, each with its own local naming and addressing scheme.

If we assume that we can build such an overlay network and that it can scale

"reasonably," we find a number of interesting benefits to the deployment of such a system as well as potential red herrings. The purpose of this work is to outline the issues and consider the tradeoffs.

Many previous approaches to providing end-to-end connectivity across middleboxes assume a core to which all forwarders are attached (46; 47; 142) or recognize that fragments can have their own address space allocation, but assume a globally unique DNS-like name for resources (23). Like Plutarch (34), PAN achieves truly separate naming and addressing in different fragments. However, unlike Plutarch, PAN does not require the boundaries between fragments to be well-defined.

1.3.3 Contrasting PAN and VPN

At a superficial level, Perspective Access Networks provide functionality quite similar to the functionality offered by Virtual Private Networks (VPNs). In particular, VPNs allow users to appear to be on a remote network, generally for the purpose of accessing resources only accessible to hosts on that network. Perspective Access Networks provide this, but they provide two other useful features as well: a *directory service* that allows users to specify the perspectives that they want by their characteristics and a *routing infrastructure* that can deliver traffic to the desired perspective even if the network is fragmented in a manner that prevents the user from communicating directly with the server providing the perspective.

First, the *directory service* provides a general method by which users can request perspectives. Users need not know in advance the particular server that provides the perspective but only a means of describing the perspective to the directory service.

As long as a means of reaching a perspective matching the specified characteristics exists, the client system will be able to use the directory service to gain access to that perspective. If we consider individual hosts that provide perspectives to be VPN servers, then even if a single VPN server or its network ceases to be accessible, the user will be able to use the same description to seamlessly build a circuit to a different VPN that provides a perspective with the same characteristics. In this way, Perspective Access Networks address the fragility that individual VPN servers may have. To some extent, the directory service may also provide some robustness against adversaries; we explore this possibility in Chapter 5.

Second, the *routing infrastructure* allows clients to access perspectives that they cannot access directly. Deployed VPN servers generally rely upon the assumption that they are accessible to everyone in the "core" of the Internet; VPN servers to which access has been filtered and VPN servers behind firewalls or NAT devices may not be accessible. In addition to providing a means of advertising perspectives throughout the network, the routing framework allows network participants to be arranged in arbitrary topologies. Though we describe some scenarios that necessitate routing in Section 3.6, we believe that most uses for Perspective Access Networks today do not require routing. Routing will become more important, however, when network filtering and fragmentation become more widespread.

We revisit the distinction between PAN and VPN in Section 5.4. In Section 5.4.1, we compare PAN to VPN in the context of practical applications, and in Section 5.4.2, we consider PAN in the context of powerful adversaries.

1.4 High-Level System Overview

We give a brief overview of the PAN architecture including a description of its components and a description of how a client uses the PAN overlay to access a remote resource.

To construct a *general-purpose* system that satisfies our requirements, we propose overlay networks consisting of *forwarders* that act as intermediaries between nodes that cannot communicate directly. We discuss the role of forwarders and their capabilities in Chapters 3-5.

A PAN forms an overlay network for transport-layer traffic. The human users of a PAN interact with ordinary Internet-aware applications, which in turn interact with a PAN client via a proxy interface. Applications treat the PAN client as a generic transport-layer proxy; this proxy may use the SOCKS (78) protocol. The PAN client uses the PAN *directory service* to determine a path through forwarders in the overlay network and then sends data from the application along that path.

1.4.1 Components

The PAN system consists of the following components:

- Resources. Resources are simply hosts that offer (possibly legacy) services to which the PAN overlay enables access.

- Forwarders. Forwarders are the nodes that make up the peer-to-peer overlay network, working to establish virtual circuits through which TCP streams flow.

- Clients. The PAN client consists of two components: (a) a proxy that serves

as an intermediary between client applications and the overlay network, and (b) a mechanism for choosing paths and establishing circuits through the overlay network.

- DIRECTORY SERVERS. The directory servers obtain information about the individual forwarders. Clients contact the directory servers in turn to obtain information necessary to route traffic to the forwarders of interest.

1.4.2 Accessing Resources

Suppose that the forwarders have organized themselves into an overlay that can forward transport-layer traffic. We stipulate that each forwarder independently generates a self-certifying identifier (83), and forwarders throughout the system refer to other forwarders using these identifiers. The key insight underlying self-certifying identifiers is that as long as the size of the identifier is sufficiently large and the sources of randomness are sufficiently effective, then the chance of a namespace collision among these identifiers within the system will be negligible.

Figure 1.1 depicts how Blossom enables an Internet host to access resources outside its local fragment. Suppose that the source (labeled `foo.source.net`) wants to communicate with a host known to forwarder F_4 as `bar.target.org`. Suppose that the source knows how to talk to F_1, and that the self-chosen ID of F_4 is `79f72ae5`.[5] Then, the source will tell F_1 to open a TCP session to `bar.target.org` as seen from F_4. The control plane (consisting of directory servers) provides F_1 with routing information indicating that F_2 is the next hop en route to F_4, so F_1 knows how

[5]We chose four bytes to create an illustrative example; actual IDs would be longer. Also, in practice we use human-readable names, mapped to self-certifying IDs by a third party.

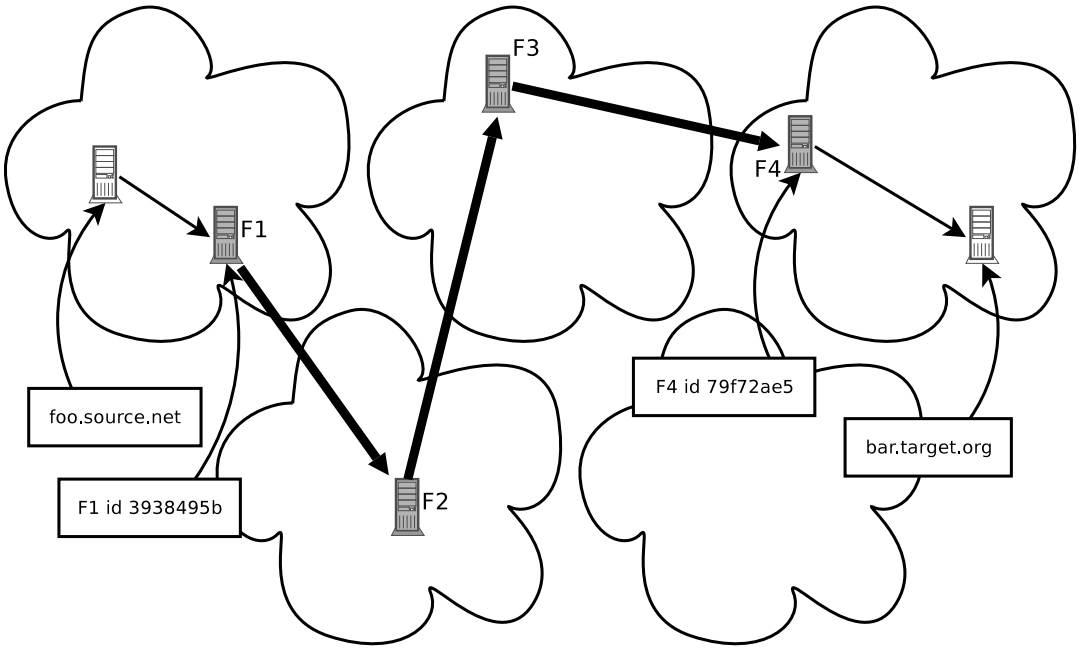


Figure 1.1: Accessing a Resource. *The source establishes a connection to* ***bar.target.org*** *from the perspective of* F_4*. DNS requests and TCP sessions are both tunneled through the infrastructure.*

to forward packets through the overlay to F_4. Next, F_1 forwards the request for `bar.target.org` through the overlay to F_4, who uses DNS to resolves it to an IP address. At this point, F_1 can tunnel the entire TCP session through the overlay to F_4. Note that this involves segmenting the TCP session—the conversation between the source and F_1 will have a different pair of source and destination addresses than the conversation between F_4 and the target resource. This means that Blossom will not work with end-to-end address-based security systems such as IPSec; we describe the policy implications in more detail in the following section.

Suppose that there are two forwarders, A and B. If some middlebox such as a firewall or NAT creates a "unidirectional link" between A and B such that A can

establish a TCP connection to B but not vice-versa, then A may establish a *persistent connection* to B that allows new paths to be built by clients in both directions.

Observe that the combined name "`bar.target.org` as seen from F_4" is globally unique, but the name was not apportioned by any authority of global scope. Also, there is no requirement that each resource be associated with exactly one forwarder; multiple forwarders may be able to reach the same resource, possibly using different names.

1.4.3 Directory Functionality

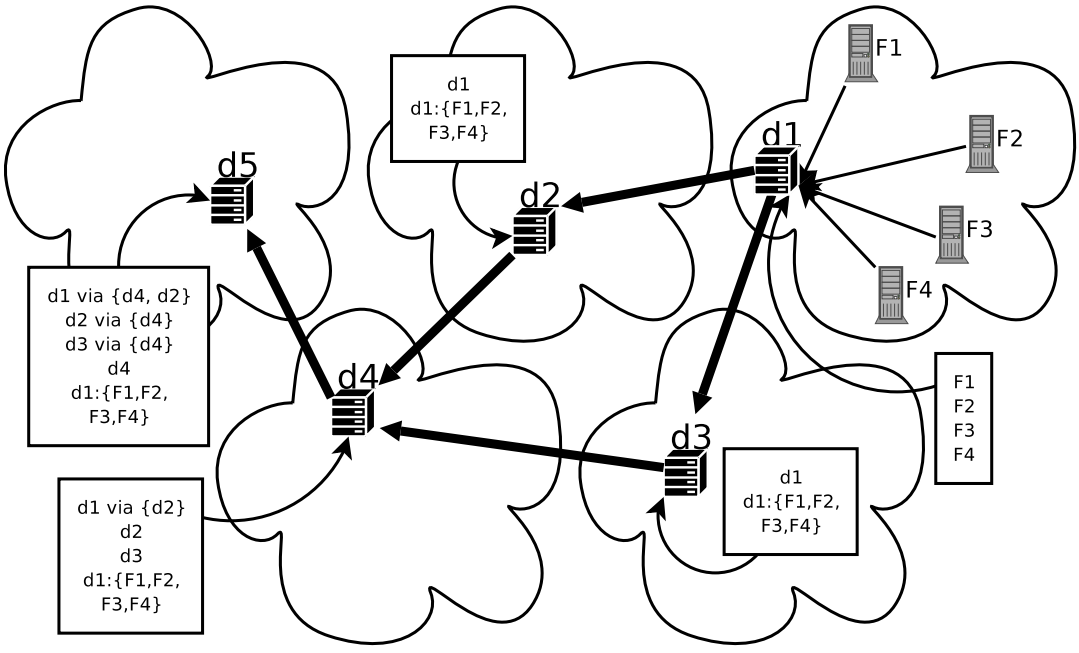


Figure 1.2: Advertising PAN Forwarders. *PAN directory servers use a* ***path-vector*** *algorithm to propagate contact information for forwarders. Black lines indicate the path taken by an advertisement initiated by the directory server labeled* $d1$.

PAN relies upon a directory service that keeps track of how to reach the various

perspectives available in the overlay network. The directory service is implemented as a set of directory servers that publish various different kinds of entries; we provide a conceptual overview:

- Forwarder Descriptor. PAN directory servers provide *forwarder descriptors* that can be used by the PAN client to establish circuits through the forwarding network. Descriptors are self-signed statements published by forwarders that contain contact information, including IP address, port, and RSA key, as well as salient information about the capabilities of the forwarder, including exit policy and bandwidth measurements.

- Forwarder Path. Suppose that a PAN forwarder publishes its descriptor to some particular directory. The PAN architecture allows forwarders to publish their descriptors in directories in locations from which those forwarders are not directly accessible. If the forwarder is not directly accessible by nodes that receive descriptors from this directory, then the forwarder must provide instructions by which some client can reach it. These instructions appear in the form of a *path*, listing a particular sequence of nodes to which to connect to establish a circuit including the target forwarder. If, in the context of Figure 1.2, F_1 had published to d_5 directly, then there would be a forwarder path entry for F_1 describing how to get to F_1 from the vicinity of d_5.

- Directory Table. Directory servers publish a list of other directory servers in the system, as accrued over time through routing advertisements. Entries for directory servers that are directly reachable are trivial, containing only the

name of the server. Other entries include a path through the set of directory servers via which the remote directory service may be reached. The first four entries in the box corresponding to d_5 in Figure 1.2 represent directory table entries.

- Perspective Attributes. Not all PAN directories publish descriptors for all PAN forwarders; however, given a set of attributes that define a perspective, a PAN directory may store information that a client can use to determine a source route to a forwarder that matches the perspective it seeks.

The directory servers propagate reachability information about individual entries (both forwarders and directory servers) in their respective databases to other directory servers throughout the system. In this manner, any client using any of the directory servers throughout the system will have a measure of assurance that its data will be routed to the requested forwarder. Figure 1.2 abstractly illustrates the process in which route information is propagated through the system. Entries are propagated using a BGP-like path-vector protocol, which includes a simple route selection protocol run at each of the directory servers.

The storage and aggregation of the multiple different kinds of attributes that describe individual perspectives makes routing in Perspective Access Networks fundamentally different from Internet routing. We describe these differences in greater detail in Chapter 4.

1.5 Design Considerations

By providing a means for bridging fragmented networks in a clear and consistent way, we hope to reduce the need for ad hoc, one-off mechanisms designed to circumvent policy restrictions. With respect to this design point, our objectives are similar to those of other overlay-based systems that we will examine in the next chapter (46; 142). Like many systems, the system we propose has potential to be used maliciously (refer to Chapter 6 for a description of the political and legal risks). While we do not wish to condone malicious use, we believe that in many cases, the circumvention of filtering mechanisms may be necessary since network access decisions are sometimes made implicitly, for practical reasons, rather than explicitly, for policy reasons.

For example, the value of the decrease in the number of requests received by call centers might be a strong incentive for a network access provider to institute a filtering rule (e.g., filter all incoming TCP connections so that everyone is protected by default). However, the value of implementing a system that allows exceptions for particular users or services may be insufficient to justify the cost of such a system, even though such exceptions may be entirely consistent with policy (e.g., there may be no policy reason why people should not be allowed to opt-out of the filtering). Perspective Access Networks may be used to defray the cost of implementing exceptions.

We do not intend Perspective Access Networks to present a means by which end users can abuse remote Internet services blocked by their network access provider, even though abuse via Perspective Access Networks is possible. Quite the contrary, we seek to make it easier for network access providers to implement reasonable policies

that are less tightly integrated with routing and filtering mechanisms. We achieve this by providing infrastructure that allows providers of PAN services to determine which resources to offer to PAN users.

PAN allows individual Internet peers to provide policy-compliant access to services without requiring modification to the configuration of the network infrastructure. We believe that our work is a testament to the ineffectiveness of network-based (walled-garden) strategies in achieving security, and we believe that the existence of such a tool encourages more widespread deployment of end-to-end security systems.

1.5.1 Objectives

Perspective Access Networks are designed to achieve several design objectives:

Objective 1: Locality

Since Perspective Access Networks allow users to specify a particular location from which to view Internet resources, it becomes possible to create resources whose content is tailored to particular locations.

Additionally, the existing Internet paradigm intends for there to exist a global namespace in which centralized authorities allocate names hierarchically and uniquely. Conversely, in the real world, the meaning of a name is dependent upon its context (unless economics dictates otherwise). That is, there can exist two companies named *Olympic*, each selling a different service (e.g., a global airline service and a pizza service in Watertown, Massachusetts). In the context of the Internet, this means:

- (a) from some locations, an observer might not be able to see a particular

resource because the observer is blocked, and

- (b) from some locations, while the observer might be able to see a particular resource, the resource itself may appear different because the service is location-specific.

A system that facilitates communication across network fragments may allow for the development of distinct local namespaces, in which names have local meaning, while also allowing access to objects in other namespaces that happen to bear the same name. This may afford businesses the opportunity to protect their trademarks, avert some Internet namespace arbitrage, and generally lead to relaxation of an unnatural constraint on naming.

Some trademarks like "Xerox" prevent others from re-using the name but only because lawyers have determined it reasonable to uphold the validity and universality of the particular trademark; for many smaller organizations, name re-use is allowed and unchallenged. Why assume that all names must be unique just because a few organizations insist that their names be unique everywhere? We would rather not take a position on this; quite the contrary, we believe that technology should not get in the way of reasonable legal process. A technology that requires global uniqueness takes the courts (and thus society) out of namespace decisions.

Thus, we abandon global uniqueness of names in favor of flexibility. For example, in Figure 1.3, there are two resources named `www.google.com` in the left and right fragments. The service provided by each resource should not be required to be the same. Instead, a host in the left fragment should be able to access the `www.google.com` resource in the right fragment via the PAN forwarder $F2$.

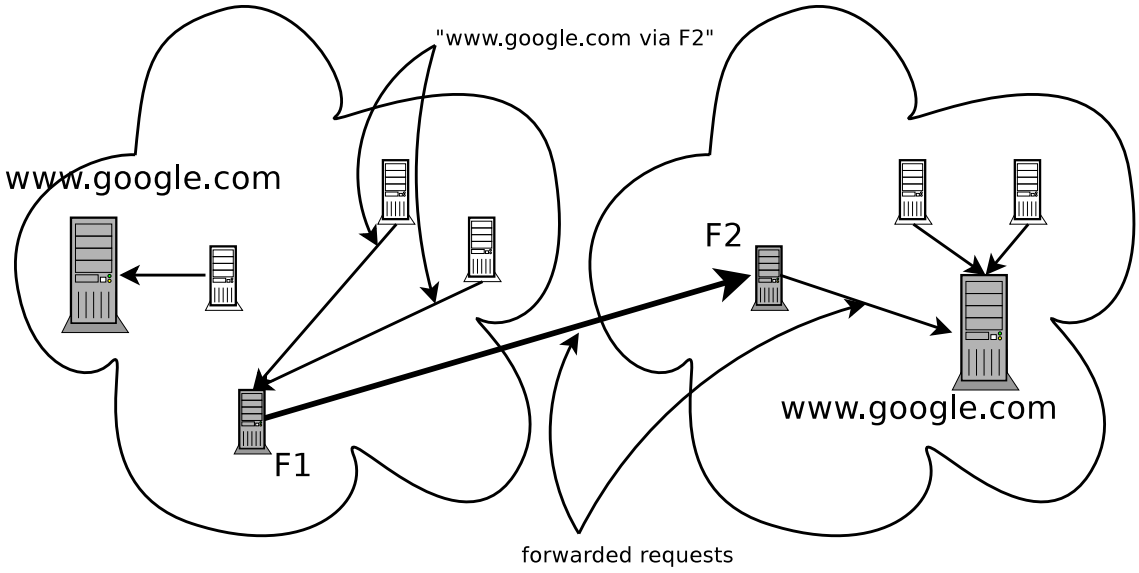


Figure 1.3: LOCALITY. *Multiple services with the same name may coexist within different local namespaces.* (Meaningful names within a local space.)

Objective 2: Access Through Obstructions

Sometimes, open communication between networks is compromised for architectural convenience rather than policy reasons (e.g., a firewall that errs on the side of filtering rather than allowing certain traffic might be deployed for convenience, and instituting exceptions for some small proportion of systems behind the firewall may be prohibitively difficult). Policy decisions must be made at some level, but technical limitations should not dictate policy.

PAN provides an architecture that facilitates the use of intermediaries to allow communication between entities that for whatever reason cannot communicate directly. In Figure 1.4, hosts on the right-hand side requesting resources located in the private network on the left-hand side should be able to access the resources, provided forwarders F_1 and F_2 can communicate and maintain a persistent connection to each

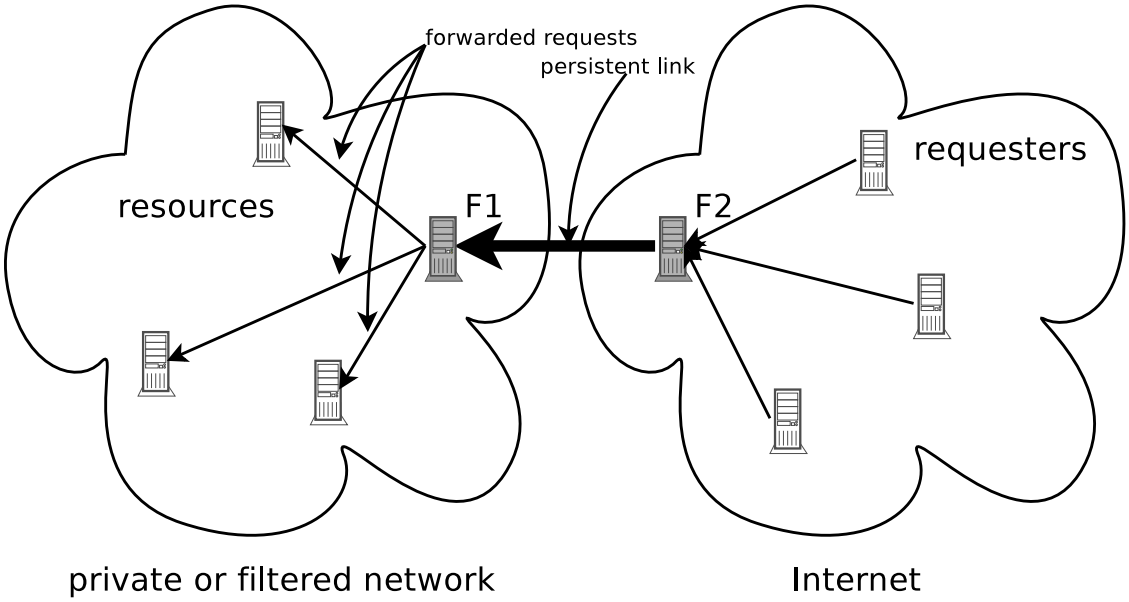


Figure 1.4: Access Through Obstructions. *If two hosts can both access forwarders within the same forwarding infrastructure, then those two hosts can use the infrastructure to communicate.* (Circumvent technical barriers.)

other. We believe that technical barriers should not implicitly set policy: we intend to circumvent *these* technical barriers, not barriers established for policy reasons.

Objective 3: Decentralized Resource Allocation

Contrary to popular belief, the Internet is not entirely a distributed network. While its management is somewhat decentralized, many aspects of its structure and governance are hierarchical in nature. Autonomous systems engage in peering relationships in a manner that promotes the set of "tiers" that characterize the organization of Internet service providers today. Both the addresses and the names used to identify resources are allocated by a collection of governance organizations, arranged hierarchically. Such an arrangement is contrary to the underlying relationships among

organizations interested in using the Internet to communicate.

PAN seeks to provide a means by which the Internet can grow without requiring the consent of third parties such as Internet service providers and DNS registrars. In particular, we want to afford users the ability to add an arbitrary namespace outside the hierarchy and then connect it to the rest of the Internet.

In Figure 1.5, a new network fragment on the left is set up to deploy a PAN forwarder called F. Adding this fragment to the existing PAN infrastructure requires only that a persistent connection be established with an existing PAN forwarder. In this case, forwarder F_1 might be chosen initially, but if F_3 becomes reachable or more convenient later, then forwarder F can set up a persistent connection with F_3 instead.

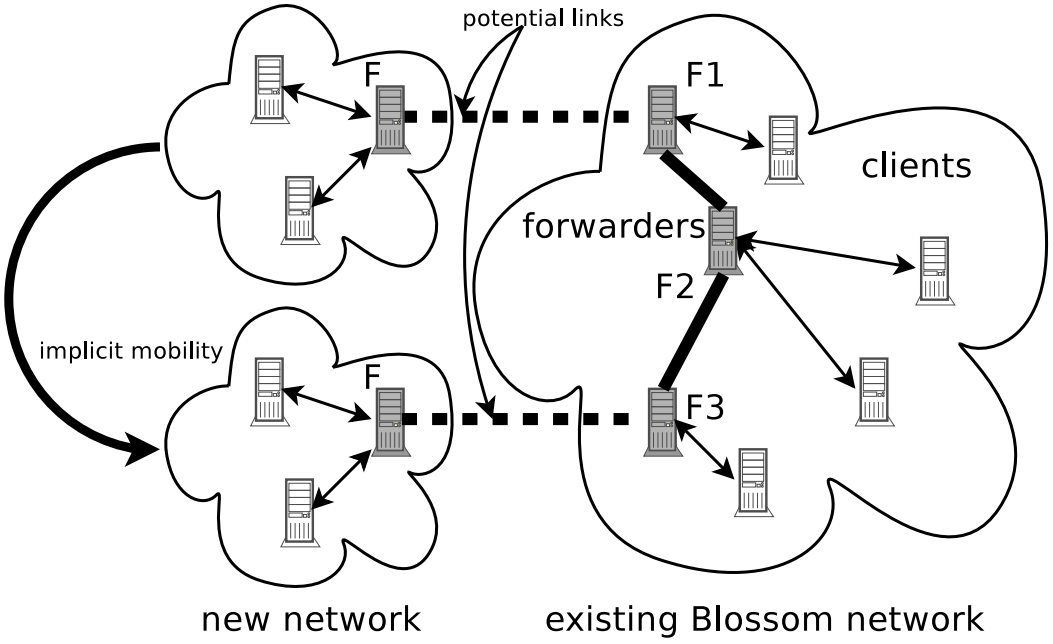


Figure 1.5: Decentralized Resource Allocation. *Adding a network and its abundance of resources to the system need not require specific allocation of names, addresses, or routing from centralized authorities.*

Objective 4: Deployability

Any complex system of sufficiently large scale that cannot be deployed incrementally will never amass enough interest to overcome the economic hurdles to deployment. PANs must provide substantial benefit even if their rate of adoption is quite limited, and PANs must be able to coexist and function without modification to existing Internet infrastructure components. In particular, both clients and servers should be able to simultaneously access both regular Internet resources and resources available through a PAN. To this end, we have developed a prototype that leverages the Tor overlay network (38) and is immediately usable by any client with no changes required to the operating system running on the host. (We suspect that a typical user of a PAN will use the normal Internet to access most resources.) An interesting consequence of running this prototype is that we can detect subtle differences in the service provided by some resources (such as Google), depending upon our choice of last-hop forwarder.

1.5.2 Tradeoffs

The deployment of Perspective Access Networks carries technical costs as well as functional benefits, as we make a number of tradeoffs to achieve our various goals:

- Loss of Control. PAN can be used to circumvent purposeful barriers, so parties with an interest in implementing purposeful restrictions might be inclined to oppose the deployment of PAN forwarders.

- New Discovery Constraints. With PAN, we will need a way to find the

forwarder that can access the remote resource that we want. We propose a global distributed directory service that has some of the characteristics of DNS, although it is not explicitly hierarchical. One significant business concern is how the provider of an Internet resource accessible from only some locations will refer to the resource when describing its location to arbitrary people. Potentially, providers of online services must use both the name of the resource as well as a description of a perspective that can reach the resource (though the set of adequate perspectives may be intrinsically defined by the nature of the service).

- New Scalability Constraints. By giving up a global unique namespace for resources, we need some way to uniquely identify a resource. For this reason, we require forwarders to generate unique, self-certifying identifiers and concatenate these identifiers with the local names of resources to uniquely identify the resources, and these identifiers of forwarders must be propagated with directory entries through the PAN overlay. Also, there seems to be an inherent trade-off between the ratio of forwarders to directory servers and the frequency of updates for particular directory entries.

  Regarding "reasonable" scalability, consider that there are serious limits to the theoretical scalability of BGP4 (109), the de facto protocol for interdomain routing, and nonetheless this system is quite functional and useful on a global scale. The propagation of routing updates through PAN follows a similar pattern. Note also that one clear alternative to propagating routing updates is performing queries (and possibly caching results); this approach introduces a different set of scalability concerns and also complicates connection setup.

The granularity with which PAN clients describe perspectives also affects scalability. If clients are allowed to specify perspectives very precisely and the network grows large, then directory servers will not be able to handle the number of entries or volume of control plane traffic. Section 5.5 presents a detailed discussion of the tradeoff between query expressivity and table size.

- New Namespace Challenges. We argue that we do not really need globally unique identifiers across all components that want to talk with the outside world, but only a way to uniquely identify resources.

1.6 Outline

The aforementioned design tradeoffs frame the discussion of our architecture and implementation. We continue this discussion in Chapter 5. We organize the remainder of the thesis into six chapters, as follows:

- The second chapter provides necessary background, including a survey of related work, an examination of the thesis in the context of extant literature, and an exploration of literature that addresses problems associated with overcoming fragmentation and providing locality.

- The third chapter conveys the design of the overlay network, including arguments supporting the design, a discussion of both technical and non-technical aspects of its applicability, and a characterization of what kinds of perspectives can be propagated through the network. We also present Blossom, our realized PAN prototype, we characterize the requirements for the transport-layer

tunnelling infrastructure upon which it relies.

- The fourth chapter describes the directory service, including its discovery mechanism, the interaction between forwarders and directory servers, and the manner in which perspective data are propagated through the network. We present a policy framework for specifying which perspective attributes and forwarder descriptors to propagate, and we argue that this framework is sufficient to meet the requirements for deployability and provider incentives.

- The fifth chapter presents an evaluation of Blossom, including both experimental results and some theoretical reasoning about performance tradeoffs. The experimental results provide insight into the central scalability tradeoffs as observed by clients, the directory service, and the network itself. In addition, we describe some strategies for implementing filtering policies and using aggregation to improve scalability. We provide evidence that the routing tables in directory servers are manageable for a set of practically useful perspective queries. We conclude this section with a discussion of factors contributing to the socioeconomic impact of Perspective Access Networks.

- The final chapter concludes by re-examining the costs and benefits of Perspective Access Networks in the context of the Internet of today. We consider the technical, social, and political implications of this tool in the context of the tussle spaces (28) and end-to-end arguments (27) presented earlier. We speculate about how the landscape might change, and we provide the groundwork for future research projects in related areas.

Chapter 2

Related Work

This chapter provides a literature search to position Perspective Access Networks in the context of prevailing work in the field. In general, we consider systems from the following areas:

- Routing: the process of moving traffic around in a network so that it reaches the correct destination, including methods for robustness against transient network malfunctions, accidental misconfiguration, or shortcomings related to slow routing convergence.

- Indirection: the method of communicating indirectly by using proxies or waypoints to circumvent systemic reachability problems.

- Interoperating with Middleboxes: either providing a means by which existing middleboxes can function without violating central Internet design principles or providing a more versatile Internet architecture in which the benefits of middleboxes can be achieved less intrusively.

- Decoupling Policy from Mechanism: approaches to improve the ability for network mechanisms to incorporate, exchange, or negotiate policy. Mechanisms sometimes implicitly dictate policy, even when actual (stated) policy differs from that which is implemented by the mechanism.

- Anonymity Networks: networks that allow participating users to obfuscate their identities or network locations.

- Covert Communication: a method of disguising traffic so that it blends in with existing traffic in a network or channel.

- Embracing Heterogeneity: the principle that an appropriate way to manage inconsistency is to create useful bridges between inconsistent components rather than impose some kind of universal organizational framework.

- Distributed Directories: any of a number of methods to improve the performance or functionality of distributed data stores, including caching and delegation.

Various systems from the literature address problems related to network fragmentation, and the design of PAN adapts aspects of their approaches to the problem of accessing content within a fragmented network. From a network standpoint, the goals of Perspective Access Networks are most similar to FARA (Section 2.4.1) and Plutarch (Section 2.7.2). From an end-to-end standpoint, the goals of Perspective Access Networks are most similar to Platypus (Section 2.4.2) and Tor (Section 2.5.1). In the sections to follow, we differentiate PAN from these systems. Table 2.1, located at the end of the chapter, provides a summary.

2.1 Routing

Routing is essential to a Perspective Access Network, since the data sent by a client must find its way through the network to a forwarder whose perspective matches the requirements specified by the client. First, we consider interdomain routing, a large-scale, policy-driven system implemented by the Border Gateway Protocol. Then, we consider routing within overlay networks.

2.1.1 Interdomain Routing

There are tens of routing protocols; they can be broadly split into two categories: *intradomain*, or internal, routing protocols, and *interdomain*, or external, routing protocols. Organizations under cohesive administrative control (companies, universities, Internet service providers) use intradomain routing protocols to exchange information about how to reach machines within their own purview. Interdomain routing protocols are used to exchange and propagate reachability information *between* such organizations. This split reflects the coarse structure of the Internet: many networks connected to each other. It also reflects the different needs and requirements for routing protocols for use in intra- versus interdomain routing. While there are several internal routing protocols in use today, there is only one interdomain routing protocol: the Border Gateway Protocol (BGP) (109; 124).

BGP views the Internet as a collection of interconnected *autonomous systems*. An autonomous system (AS) is a portion of the network under single administrative control (at least as far as routing is concerned). Each AS connects to other ASes; the

routers in each AS that connect to their counterpart in other ASes are called *border* routers. These neighboring border routers connect *directly* to each other, that is, there are no routers between them. (This is not strictly true, nor is the assertion that only neighboring routers speak BGP to each other, but the details are beyond the scope of this discussion.) Over this direct connection, border routers establish *BGP sessions*; there may be many BGP sessions over each link, but there are (almost) never BGP sessions between non-neighboring routers. BGP sessions are used to exchange network reachability information—each router tells its neighbor what address ranges (also known as address prefixes, or just prefixes) to which it knows how to route traffic, along with ancillary information that is used to make the decision of whether this router will actually be used to route that part of the address space.

As BGP provides information for controlling the flow of packets between ASes, the protocol plays a critical role in Internet efficiency, reliability, and security.

Two of the most significant concerns facing modern interdomain routing are protocol oscillations and security vulnerabilities (43). Section 5.2.3 describes how the PAN directory service compares to BGP with respect to these issues.

Like the distribution of routing information within BGP, the distribution of reachability information within Perspective Access Networks may potentially grow to large scales, and distribution points for such information will be operated by parties with an interest in specifying policy. Chapter 4 illustrates a means by which individual PAN directory servers may specify local policy, and Chapter 5 describes a number of approaches for promoting scalability and resource management.

2.1.2 Overlay Routing

Andersen et al. (5) propose the use of Resilient Overlay Networks (RON) to address certain limitations of the interdomain routing protocol BGP (109), including (a) slow recovery from failures, (b) insensitivity to specific requirements of applications, and (c) insufficient flexibility in supporting policies.

RON has three goals: (a) provide additional robustness in the event of localized network malfunction, specifically recover from malfunctions faster than BGP, (b) provide tighter integration with applications to allow them some control over the underlying routing, and (c) provide the ability to express more complex policies than those that can be expressed via BGP. RON provides an overlay infrastructure that participating nodes within the Internet can use to attain these additional benefits. Like PAN, RON aims to overcome network obstructions. However, its purpose is essentially limited to finding alternate routes more effectively than BGP. Thus, it does not address our interest in locality or decentralized management.

In essence, RON is a response to several of the shortcomings of BGP, namely (a) slow recovery from failures, (b) insensitivity to specific requirements of applications, and (c) insufficient flexibility in supporting policies. As the RON authors note, BGP avoids providing these benefits in the interests of scalability. The scalability of RON is fundamentally limited by its design, but the question remains whether it will have benefit to smaller communities who want to achieve robustness and policy benefits within their local group.

Like our proposed system, RON aims to overcome network obstructions. However, as described in Section 1.1.2, Perspective Access Networks do not (in general) attempt

to handle the transient routing failures addressed by RON. While RON may compensate for the shortcomings associated with slow BGP convergence, PAN compensates for long-term unreachability, such as that imposed by policy or filtering mechanisms. In addition, we believe that PAN is more scalable than RON; we provide evidence for its scalability in Chapter 5.

2.2 Indirection

The PAN architecture is designed to create a general means of providing access to services that are not accessible directly. There exist a few approaches to bridging regions of the network that are not directly connected that have been proposed.

2.2.1 Internet Indirection Infrastructure

The Internet Indirection Infrastructure (I3) (125) provides a "rendezvous-based communication abstraction" in which providers of services advertise to a particular location in the network, and those peers requesting services communicate with that location rather than with the provider directly. Indeed, services like anycast (103), multicast (37), and mobility (119; 120) all require some measure of "indirection". I3 offers a standard substrate upon which all of these can be built and provides mechanisms for achieving composition of services, scalable multicast, etc., which have tangible benefit in the real world. The authors present how the functionality of various existing systems for providing these services can be achieved with I3.

Finally, I3 provides useful delegation primitives that serve as inspiration for DOA, which we discuss in Section 2.3.

Services in I3 are registered with the infrastructure, whereas in PAN, only the perspectives are registered with the infrastructure, and a client can use a perspective to access any resource that a perspective can contact directly (provided that the host offering the perspective allows such access).

2.2.2 TRIAD

Systems for "content routing" often employ overlays to organize content logically and providing suitable naming infrastructures to enable a means of accessing arbitrary resources (20; 60). TRIAD (23) characterizes the Internet as a set of regions with local addressing, arranged such that some peers have access to multiple regions. The authors justify this characterization by noting the preponderance of NAT boxes. Peers with access to multiple address spaces use a protocol called WRAP to relay content between different regions of the network in a stateless manner. As packets pass between different address spaces, a middlebox bridging the two spaces modifies the addresses in the packet headers.

TRIAD uses globally unique hierarchical, DNS-style names to identify networks, and the authors propose a modified BGP to propagate suffixes for these names, rather than prefixes for IPv4 addresses. This modified interdomain routing system provides support for aggregation based upon names (by "suffixes" rather than "prefixes"). The system raises questions about scalability, since the physical location of domains within the network topology of the current Internet is arguably correlated much more closely with address ranges than with domain names, and the number of distinct domain names immediately descended from top-level domains far exceeds the number

of prefix entries in existing BGP routing tables.

TRIAD could potentially provide a means by which networks could be connected to each other in arbitrary topologies independently of a central authority to govern addresses. However, TRIAD still relies upon the idea that resources must be universally named, and to this end the authors propose a complex protocol to facilitate routing according to these names. A later paper by the same authors analyzes the feasibility of TRIAD (60). The empirical analysis ignores hardware performance inside TRIAD-enabled routers, presuming that network bandwidth is the limiting factor. Also, the argument for the degree to which aggregation of names is possible and efficient seems insufficiently strong.

TRIAD uses globally unique, hierarchical names to identify networks; these names are propagated throughout the system via BGP-like advertisements among TRIAD nodes. In PAN, names of resources need not be globally unique, and names of PAN forwarders are non-hierarchical. TRIAD also requires the middleboxes themselves to participate in the bridging infrastructure; the PAN architecture does not.

2.3 Interoperating with Middleboxes

Network-layer intermediaries (that is, middleboxes) exist for important reasons and we have every reason to believe that these reasons will continue to prevail in the future. Middleboxes are used to solve three problems, and these problems are unlikely to change substantially in the foreseeable future: (a) bridge IP address spaces (e.g., NAT/NAPT), (b) discard unwanted packets (e.g., firewalls), and (c) improve performance (e.g., caching, load balancing). The authors argue that it is difficult and

costly to administer a network host, and middleboxes help alleviate some of the risks and complexities.

This section prevents various approaches to the problem of identifying and accessing resources through middleboxes.

2.3.1 Host Identity Protocol

The Host Identity Protocol (HIP) (90) provides unique identifiers for each communication endpoint, thus creating an endpoint identifier that is independent of network location. The idea that every Internet entity can be specified by a unique network identifier forms the basis for numerous other projects, including DOA (described in Section 2.3.2).

HIP does not provide a sufficient means of actually locating the endpoints: without some sort of directory infrastructure, we are left with querying and broadcasting, both of which are inefficient. We believe that building the directory services constitutes an interesting technical challenge, which is a key focus of our work. Also, since HIP does not provide a means by which we can name existing, "legacy" services, every service that can be designated using HIP must itself be an active participant in HIP. Finally, since the content of each packet must be encapsulated within a HIP datagram, we need to either (a) change the protocol stacks at the edges or (b) establish an infrastructure for tunnelling.

Unlike PAN, HIP creates new identifiers for the transport-layer endpoints, requiring modification to the protocol stack.

2.3.2 Delegation-Oriented Architecture

The goal of Delegation-Oriented Architecture (DOA) is not so much to argue that policy should be pushed to the edges of the network, but rather to describe how to create an Internet architecture that allows middleboxes to perform their functions without violating two fundamental principles of Internet design. , stated much more clearly than in the earlier Balakrishnan et al. paper: (a) every Internet entity can be reached via the use of a unique network identifier, and (b) network elements should not violate the principles of layering (in the context of this paper, this principle is essentially the end-to-end argument).

The authors argue in favor of inserting into every packet the globally unique identifier (such as that offered by HIP) corresponding to the source and target endpoints; the authors presume that the network will be able to forward replies to the source by the same method used to forward the original message to the destination. The authors also argue in favor of using delegation to allow nodes to express how they can be reached by others. Nodes wanting to access other nodes identify them by an endpoint identifier (EID), which resolves to either another EID, a list of EIDs, or an IP address. Such resolution requires an infrastructure, and the authors propose using a distributed hash table (DHT) for this purpose. Clearly, this choice introduces a number of concerns in the area of scalability, overhead, complexity, management, and flexibility, and it might be interesting to consider alternative solutions. Refreshes to the DHT must occur with a certain regularity, which constitutes overhead that may be unreasonable for routers. Also, the authors suggest the use of hint fields to improve performance, but they do not explain the extent to which such use will

be necessary in order to achieve the desired goals. As with TRIAD, IP addresses are modified in the headers as packets pass between IP networks with incompatible address spaces. However, because DOA allows resolution of EIDs to other EIDs, the authors argue that the advertisement problem observed in TRIAD (for which a modified name-oriented BGP is proposed) is essentially abstracted away.

One of the interesting observations made by the authors is that today, users are "typically stuck with whatever middleboxes lie on their path." DOA would allow users to choose their middleboxes and know when middleboxes are in use. Firewalls could become a tool that end users can explicitly configure; for example, perhaps firewall information could be auto-configured using DHCP. The most interesting implication is that functionality of a firewall will become orthogonal to topology: for example, a node could choose to use a firewall by using an EID that would address packets to the firewall, which could process them and pass them along to the ultimate destination. From this principle, one could even imagine deriving business models for firewall service; the authors included a section describing such a service. It would be interesting to examine whether, given that out-of-band firewalls could exist, organizations would still have valid reason to impose restrictions on hosts within their networks.

Unfortunately, there are some substantial deployability concerns with DOA. First, modifying the TCP and UDP pseudo-checksums as advised to support DOA would require significant modification to the IP stacks of both clients and servers. The authors spend a section describing an architecture of Network Extension Boxes (NEB), which would replace NATs in the DOA paradigm. This is one of the most essential

applications of DOA, and it relies upon a potentially complex set of messages in the control plane and some (not fully specified) global lookup service. Similarly, the DHT functionality central to DOA relies upon all DHT nodes being in the same transport domain, which implicitly requires a well-known core. This raises questions of whether nodes in the core have to be specially configured, whether they know that they are in the core, and of course whether the Internet needs to have an inherent hierarchy in order for NEBs to function. (The authors proceed to analyze three different approaches to implementation, optimizing for different tradeoffs in the space of sender computation, NEB computation, and NEB state.)

There are a number of important differences between PAN and DOA. In particular, PAN aims to provide access to remote resources without the need to modify protocol stacks. Furthermore, PAN explicitly avoids requiring either all-pairs reachability or making any assumptions about designation of a particular transport domain as the well-known core.

2.3.3 Unmanaged Internet Protocol

A position paper introducing Unmanaged Internet Protocol (UIP) (46), which aims to restore end-to-end connectivity to the Internet by establishing a system for routing based upon globally unique names chosen by the hosts themselves rather than assigned by a central authority. The system leverages distributed hash tables to provide routing based upon the names, which are chosen to be self-certifying and topology-independent.

The authors acknowledge the fundamental problem associated with naming: while

a hierarchical assignment of addresses effectively provides efficiency and scalability, the same hierarchy creates inflexibility at the edges of the network. Specifically, there exist problems with mobility (i.e., changing location in the topology requires changing address), allocation (i.e., obtaining an address requires special dispensation from the management of the hierarchy), and consistency (i.e., everyone must believe in the same hierarchy), among others.

The DHT that UIP uses requires all-pairs universal connectivity among nodes, but the authors want to support any topology. The solution they propose involves using a recursive technique to effectively generate a source route to any possible the destination, thus allowing universal connectivity. The details of this argument are not fully clear, and it remains to be seen whether this approach to maintaining universal connectivity as required by the DHT is actually efficient and functional in practice, particularly when the topology changes frequently and many nodes are unreliable.

While UIP provides a step in the direction of universal access and distributed management, its goals are different from those of PAN. First, UIP concerns itself only with identification of UIP-enabled resources, rather than accessing existing resources using UIP. Second, UIP does not address locality issues as we define them; it aims to create a single, flat Internet space rather than promote the idea of separate views of the space.

2.3.4 IPNL

IPNL (47) adds an overlay layer above IPv4 that would be routed by NATs and makes use of Fully Qualified Domain Names as end system identifiers in packets. Like

PAN, IPNL intends to provide end-to-end connectivity across NATs. Unlike PAN, IPNL allows its routers to remain stateless. However, IPNL is site-centric, requiring special configuration and deployment of "frontdoors" that connect independently managed networks to an established core. PAN makes no such assumptions, instead requiring only that there exists a forwarder capable of reaching the target network and that that forwarder has the ability to bidirectionally communicate with another forwarder in the PAN overlay. Also, PAN does not require any changes to the operating systems of end hosts.

2.4 Decoupling Policy from Mechanism

Policy and mechanism are often tightly intertwined, and sometimes mechanism itself imposes policy. For example, the inability for a client to identify a resource by name may prevent the client from accessing the resource. Overly-broad firewall rules might be easy to implement while exceptions and the concomitant accounting infrastructure might be difficult, even if stated policy allows such exceptions. We consider a few projects that incorporate approaches to separating policy from mechanism. Separating filtering policy from filtering mechanism is a central design objective for Perspective Access Networks.

2.4.1 FARA/NewArch

The FARA proposal (26; 27) specifies a general framework for decoupling identity from network location. FARA aims to provide associations between peer nodes without requiring that all entities share a common, global namespace; in this sense,

its goals are similar to ours. FARA makes use of "forwarding directives" to establish rendezvous points through the infrastructure between a source and a destination; a shim protocol between IP and the transport layer used to support this functionality is reminiscent of TRIAD. FARA is structured so that discovery may be handled by higher layer services; the location of an entity is defined by the forwarding directive, which may be obtained via the rendezvous mechanism or a FARA directory service, for example. By not requiring all entities to share a common, global namespace, one can argue that FARA takes a step toward our goals of distributed management and locality. However, the authors do not seem to envision this possibility in their test implementation, M-FARA, which avoids the "complexity" associated with dealing with an unstructured Internet by relying upon a well-known Internet core.

In addition to being largely unspecified, FARA requires modification to existing protocols and applications. The circuit discovery process in PAN may be considered a natural extension of the FARA forwarding directive.

2.4.2 Platypus

Snoeren and Raghavan (121) argue that routing policy should be enforced on the forwarding plane rather than on the control plane, as it is done today with BGP4. The authors propose a new routing architecture, Platypus, which uses loose source routing (LSR) to allow fine-grained, policy-aware route selection by the sender.

In Platypus, autonomous systems advertise all available routes, irrespective of policy, along with "network capability" metadata. Loose source routing information would be included in each packet, allowing end users to take advantage of Platypus

directly. Also, routers within the network could use the metadata to improve route selection. Most notably, this work presents a major paradigm shift: instead of requiring that local policies dictate all routing, propagate advertisements deeper into the network so that hosts and networks can make more informed decisions. Let those who need to use the route make the routing decisions, and rely upon filtering techniques to guarantee that routes incompatible with policy are not used.

However, there are a number of serious flaws with this approach. First, we have the fundamental question of whether such measures are actually useful. The authors claim that certain desirable end-to-end polices "require the composition of multiple local policies," but they fail to provide an example, let alone an empirical description of the nature and prevalence of such situations in the Internet today. Are there actually any cases in which ISPs could benefit substantially from deriving richer policies via composition? What's more, it ultimately depends upon ISPs being on board, willing to advertise routes that they themselves would prefer not to use. Perhaps the justification is that ISPs can filter non-compliant traffic, but ultimately, if an ISP does not want to forward traffic in a particular direction, it has no reason to advertise such a possibility.

Another problem with the Platypus approach is that it does not take into consideration the computational and storage constraints facing routers. The authors argue that export of all possible routes could provide more alternatives, but exporting all possible routes (a) increases network overhead by expanding the set of advertised routes, (b) increases storage constraints at each router by prescribing that it should store (and forward) the advertised routes, and (c) increases computational constraints

at each router if it is required to make decisions based upon network capabilities meta-data or dynamically choose among several possible routes on a per-packet basis.

The authors' intended scheme for authentication of routes and capabilities relies upon many secrets – which means some sort of key infrastructure. Arguably this is complex and is difficult to scale. Also, present billing systems may be unable to handle the complexity of arrangements associated with charging particular principals for use of capabilities; the authors do nothing to describe the business implications of the management constraints. Even more striking is the fact that the proposed cryptographic system used allows replay attacks; the authors acknowledge this but provide only a weak solution.

Unlike PAN, Platypus relies upon cooperation from intermediary ISPs. In PAN, we assume that if an ISP does not want to forward traffic in a particular direction, it has no reason to do so and no reason to advertise such a possibility either. However, PAN presents an argument for separating network access policy from technical decisions made at the network layer. If two PAN forwarders are both connected to the same PAN overlay, then technically speaking, each could have access to whatever the other can see, regardless of what lies between.

2.5 Anonymity Networks

Anonymity networks seek to separate routing information from identity, with the following goals in mind: (a) communicating parties will not be able to identify each other based upon their network location, and (b) the network itself will not be able to determine that two parties are communicating. In the examples we consider, these

goals are achieved by the deployment of an overlay network that carries traffic along a multi-hop path between the source and the destination.

Since the PAN architecture also requires an overlay network that carries traffic along a multi-hop path, there are important structural similarities between Perspective Access Networks and anonymity networks.

Anonymity networks can be used for anti-censorship purposes, specifically to circumvent local restrictions on access to resources. However, since the Internet is not entirely flat, the resources to which a user of these networks (or of Psiphon) has access may vary as a function of the particular overlay node (or Psiphon host) that is used as the last-hop proxy. For example, requesting a particular web page from an anonymity network might yield content that has been tailored to the particular local network or geographic region in which the last-hop proxy resides. If anonymity is the goal, then a larger anonymity set may be worth the cost of some probabilistic variation in content reachability. PAN takes the opposite approach, choosing to use an overlay proxy network to tailor content reachability, possibly at the expense of anonymity. In particular, PAN clients require the ability to specify a path based upon what resources they want to access; obfuscation of their identities is not required.

2.5.1 Tor

Tor (38) is an anonymity network derived from the original Onion Routing project sponsored by the US Naval Research Laboratory (53). Onion routing works by having a sender specify a chain of n proxies within the network, such that data will traverse each of the n proxies in sequence en route to the specified recipient. To ensure

that datagrams take the correct path, the client encrypts the message several times, starting with the most distant proxy in the chain, each time including the address of the next hop along with the ciphertext created in the previous iteration. The successive layers of encryption shape the "onion" analogy: each successive proxy "unravels one layer of the onion" to expose the identity of the next proxy to which to forward the datagram. The result is that, in theory, each proxy in the chain knows nothing about the chain itself other than the identities of the previous proxy and the next proxy in the sequence.

Tor operates as a transport-layer proxy, providing some enhancements over onion routing as originally described. In particular, Tor forwards entire TCP streams, not individual IP packets, through its overlay network. Tor manages this by having clients construct *circuits*, one hop at a time, using a method with security properties similar to conventional onion routing. Once a circuit has been established, TCP streams may be "attached" to the circuit. More than one TCP stream can share a circuit, and individual links between proxies in the overlay may carry traffic for multiple circuits.

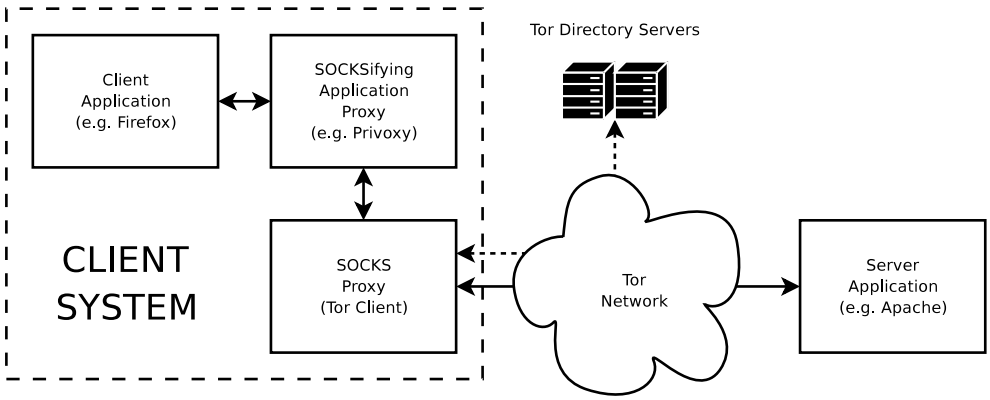


Figure 2.1: Client Perspective Diagram: Tor. *How the components of Tor are organized, from the perspective of a client.*

Tor uses SOCKS (78) as an interface to its network of forwarders that carry arbitrary TCP traffic as well as DNS requests. Tor uses a set of directory servers that publish *descriptors* for individual forwarders. A descriptor carries all of the details that a client needs to make use of a forwarder in building circuits, including its identity, its public key, its IP address, its TCP port number, and some statistics. The descriptor also provides the *exit policy*, which specifies the set of IP address and TCP port ranges to which a forwarder is willing to provide access as an exit node. Client applications send datagrams to the SOCKS proxy interface of the Tor client, and the Tor client uses descriptors obtained from the directory server to build a random path through the Tor network to the application server it wishes to contact (refer to Figure 2.1).

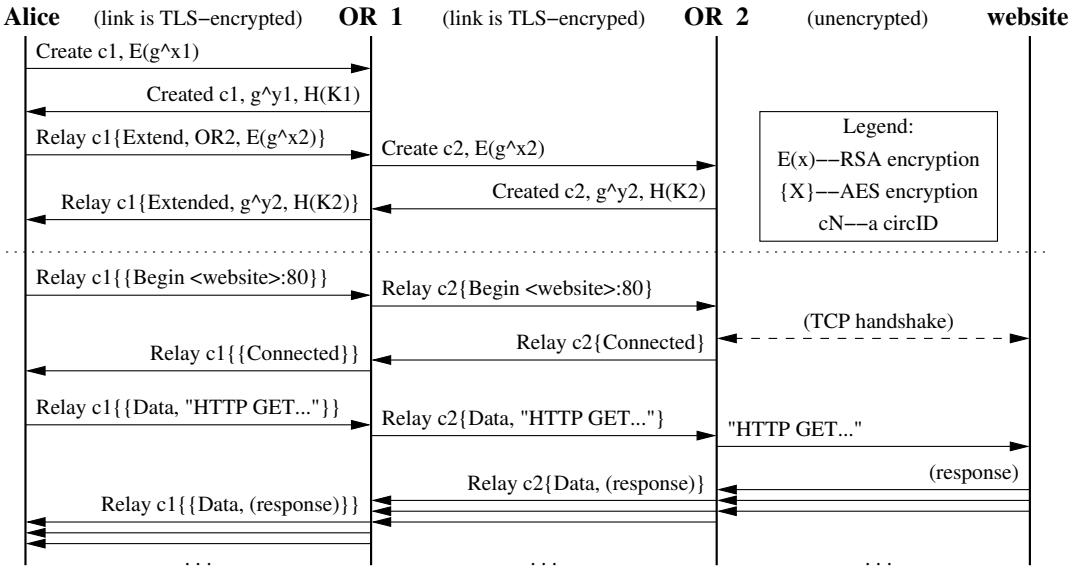


Figure 2.2: Circuit Establishment in Tor. *Circuits in Tor are extended one hop at a time, with a single end-to-end round-trip required for each extension. (This diagram is reprinted with permission from the authors of Tor.)*

Tor clients manage the construction of circuits, randomly selecting source routes

through the overlay network and extending individual circuits one hop at a time according to the chosen source routes (Figure 2.2 provides an illustration). When the Tor client receives a TCP stream from an application, it "attaches" the stream to an appropriate circuit by having the last hop (the "exit forwarder") of the circuit perform a TCP handshake with the remote server. Once the handshake has been completed, the client may communicate with the remote server via the circuit.

Tor also provides *hidden services*, which are location-hidden servers that can be accessed by clients via self-certifying identifiers.

The ability to have traverse a path through the network to a specified exit point is an essential requirement of PAN, so the Tor architecture presents a useful framework upon which to build a PAN implementation. Our test implementation, Blossom, uses Tor for circuit-building and data transport; the details of how Blossom uses Tor are presented in Section 3.4.

2.5.2 ANON

ANON (76) is similar to Tor in that it too uses onion routing to separate network location from identity. However, unlike Tor, ANON operates at the network (IP) layer, so individual packets (rather than entire end-to-end streams) are forwarded independently through the infrastructure. Unlike Tor, ANON uses link-padding techniques to provide some protection against timing attacks and rate-limiting to provide some protection against denial-of-service. This approach significantly reduces the throughput capacity of ANON, so it can only be used for signalling and other low-bandwidth applications. The potential to support low-latency, high-bandwidth

applications makes Tor a more appropriate choice for the kinds of applications that interest us, but it is entirely conceivable to build a Perspective Access Network that uses ANON as a forwarding substrate.

2.6 Covert Communication

Sometimes, disguising the identities of the communicating parties is insufficient; evading discovery may require hiding the fact that communication is taking place at all. Traditionally, this is the realm of *steganography*, the practice of ensuring that the existence of a message is known only to the intended recipient. We refer to a medium capable of carrying a secret message without exposing its existence as a *covert channel*. We do not provide a treatment of steganography or covert channels here; refer to the whitepaper by Johnson and Jajodia for an introduction (70).

2.6.1 Psiphon

Psiphon (68) is a proposed[1] single-hop proxy application used to circumvent content filtering. A host outside the filtering regime installs the Psiphon proxy software, and remote hosts that are connected to the Internet via networks controlled by the filtering regime can use the proxy to access blocked web sites. Psiphon is a personal (rather than general-use) circumvention tool, which means that while Psiphon users must establish out-of-band trust relationships with parties on the other side of the filtering regime, Psiphon offers some degree of protection against the threat of an adversary enumerating the list of proxies.

[1]Psiphon (Frequently Asked Questions), `http://psiphon.civisec.org/`

While PAN provides no explicit means of establishing social networks and authenticating parties based upon reputation or status within the social network, PAN forwarders certainly have the option of refusing to extend circuits for any reason, and failure to authenticate via some out-of-band mechanism could potentially be a perfectly valid reason.

2.6.2 Infranet

Infranet (44) is an anti-censorship system in which various web servers distributed throughout the Internet cooperate to provide a covert channel through which users can access censored web resources. The idea is that traffic sent through the covert channel will appear to be ordinary web traffic, and users of the channel will have plausible deniability about their participation.

Perspective Access Networks do not provide covert channels. In theory, PAN traffic could be sent over covert channels, and doing so may enhance the usefulness of PAN. For example, certain PAN instances (e.g., those designed to allow dissidents to access content from deep inside oppressive regimes) may benefit from the secrecy that covert channels provide.

2.7 Embracing Heterogeneity

Much of the literature about middleboxes and network fragmentation focus upon means of mitigating the problems associated with middlboxes, working around inconsistencies among networks, and generally incentivizing Internet participants to play by the rules of some global system in which consistency prevails. However,

some projects present intriguing arguments in favor of the principle that perhaps the Internet should not be totally flat after all.

2.7.1 Semantic-Free Referencing

Semantic-Free Referencing (141) stipulates that resources have globally-unique "semantic-free tags", high-entropy bit strings perhaps generated as self-certifying names by the resource provider. A client would use the semantic-free tag rather than a hostname to identify the website, and a Reference Resolution Service (RRS) would map human-readable names to semantic-free tags. The goal is to decouple the name of a resource from its content; note that this is subtly different from the *naming locality* goal of PAN. The possibility of having multiple different RRS servers suggests that this approach could lead to a form of locality, since different local regions or classes of organizations could use different RRS servers to canonicalize human-readable names. The authors provide little discussion of how multiple RRS servers could conceivably exist in practice, or why a single RRS infrastructure similar to DNS would not emerge, other than to suggest that there could be a competitive market.

Indeed, the value of DNS hostnames is apparent from the many costly disputes associated with namespace contention. (Parenthetically, use of HTTP virtual hosts is one way in which web servers separate their content from their network-layer address; the HTTP Host field allows the same server to house websites corresponding to multiple DNS hostnames, each with its own distinct set of files and configuration parameters.) The Web is undoubtedly a contributor to this phenomenon, since in-

dividual websites are identified by their DNS hostnames. It may be useful for Web content providers to separate their websites from their DNS names, and it may be useful for owners of DNS hostnames to be able to separate the names from Web content in order to avoid expensive disputes.

To address this problem, the authors propose "semantic-free referencing" (SFR), which is a means of providing globally unique names in the form of "semantic-free tags", high-entropy bit strings perhaps generated as self-certifying names by the website owners themselves. It is possible to imagine a "search-engine only" world without DNS, in which a client might obtain a semantic-free tag for a website from a search engine and subsequently use this tag rather than a hostname to identify the website, and a field for this tag could be used in place of the HTTP Host field. Since these tags would not have semantic meaning to humans, the authors propose a Reference Resolution Service (RRS) to map human-readable names to semantic-free tags.

The possibility of having multiple different RRS servers suggests that this approach could lead to a form of locality, since different local regions or classes of organizations could use different RRS servers to canonicalize human-readable names. The authors provide little discussion of how multiple RRS servers could conceivably exist in practice, or why a single RRS infrastructure similar to DNS would not emerge, other than to suggest that there could be a competitive market.

Not only does PAN aim to provide locality to Internet services in general rather than exclusively the web, but the approach PAN takes to locality is quite different. PAN still relies upon regular DNS, but allows the DNS hierarchy to be different from the perspective of each forwarder. While this does not provide the same flexibility as

SFR, it mitigates some of the same concerns.

2.7.2 Plutarch

Plutarch (34) takes the leap of considering network fragmentation as the inevitable result of political or economic forces rather than some technical obstacle to be overcome. The authors convincingly argue that avoiding global management would promote innovation. Like PAN, Plutarch does not require a well-defined Internet core or global names. Plutarch "contexts" are similar to the "fragments" that we describe. However, like IPNL and unlike PAN, Plutarch requires these contexts to be well-defined and non-overlapping. Moreover, Plutarch requires special configuration of middleboxes that serve as the boundaries between contexts. Plutarch also resolves names via a peer-to-peer search, which PAN avoids that approach in favor of reducing overhead and improving connection setup time.

2.8 Distributed Directories

The directory in PAN is distributed among a potentially large number of individual directory servers, which perform not only a routing function analogous to BGP participants but also a lookup function by which they provide information to clients so that they can select a path through the infrastructure. Delegation and caching methods can improve scalability and performance, so we consider how these methods are applied in the context of existing systems.

2.8.1 Domain Name System

The Domain Name Service (87; 88) is the widely used directory service for resolution of hostnames and IP addresses in the Internet. DNS names are constructed and resolved, and updates are propagated across DNS servers in a hierarchical manner. The PAN forwarder ID space is flat because forwarders use self-generated, self-certifying identifiers. This means PAN directory servers can neither take advantage of the hierarchical approach of DNS nor can perform aggregation of forwarder identifiers as they propagate forwarder information through the directory service. The latter approach is that used by BGP (124), which aggregates prefix information to reduce the number of entries BGP has to carry and store. We explore the design tradeoffs that arise from our approach in Chapter 3.

2.8.2 Filesharing Networks

Peer-to-peer file sharing systems dominate Internet traffic today. These systems require functionality that allows peers to resolve files (or file attributes) of interest to IP addresses of hosts that store the files. Some peer-to-peer systems use a centralized approach to providing this lookup functionality. For example, Napster placed the entire index of (filename, IP address) mappings on a single host. Apart from the potential scalability concerns, this approach assumes clients can access the centralized index. In PAN, we build our directory service taking into account that the Internet is fragmented and not all clients can necessarily reach one single directory server. Distributed Hash Tables (DHTs) (such as CAN (107) and Chord (126)) distribute this load across the participating peers. DHTs tightly control both the placement

of mappings on peers and the overlay topology which allows the efficient lookup of mappings. DHTs also assume that peers will be able to bidirectionally communicate with the peers that have been assigned to be their neighbors barring transient network partitions. Finally, *unstructured* peer-to-peer file sharing networks, such as Gnutella[2] provide an "ad hoc" directory lookup service in that lookup queries flood the network in search of a peer who may have the mapping of interest. PAN is designed with the goal of minimizing connection setup latency for clients connecting to arbitrary services. Thus, clients do not request forwarder information via flooding because connection set up latency would grow quickly with population size. In contrast, file-sharing networks, minimizing the lookup time is not of priority because file download time dominates lookup time.

2.8.3 Cooperative Web Caching

Various systems been proposed to allow groups of participating caches to track what web objects are cached at what proxies and to exchange cached web content amongst themselves. The overall goal is to bring a particular web object to the cache that is closest to the clients requesting that web object. Previous proposals include hierarchical cache schemes (e.g., (21; 71; 143; 31)), hash-based schemes (71; 136), directory-based schemes (42; 86; 130), and multicast-based schemes (e.g., (131)). All of these schemes assume that any proxy participating in the cooperative caching scheme can communicate bidirectionally with any other proxy; PAN does not have this option.

[2]Gnutella Protocol Specification, `http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf`

System	*Categories*								*Characteristics*						
	Interdomain Routing	Indirection	Interoperates with Middleboxes	Decouples Policy from Mechanism	Anonymity	Covert Communication	Embraces Heterogeneity	Distributed Directories	Modifies Protocol Stacks	Requires ISP Participation	Assumes an Internet Core	Access to Legacy Resources	Stateless Forwarding	Perspective-Based Approach
BGP	×									×	×	×	×	
RON	×								×		×	×	×	
I3		×							×		×		×	
TRIAD	×	×	×						×	×	×		×	
HIP			×						×	×	×		×	
DOA		×	×						×	×	×		×	
UIP		×	×						×	×	×		×	
IPNL	×		×						×	×	×		×	
FARA/NewArch	×	×		×				×	×	×	×		×	
Platypus	×			×						×	×	×	×	
Tor		×			×			○			×	×		
ANON		×			×				×		×		×	
Psiphon		×				×					×	×		
Infranet		×				×					×	×		
SFR							×	×			×	×		
Plutarch	×						×		×	×			×	
DNS								×		×	×	×		
P2P Filesharing		○			○			×			×			
Web Caching								×		×	×			
VPN		×		×							×	×		
PAN	×	×		×			×	×				×		×

Table 2.1: Summary of Related Projects. *A marked cell indicates that the system has the given property: a cross (×) denotes* ***always****, and a circle (○) denotes* ***partially, optionally, or under some circumstances****.*

Chapter 3

Network Architecture

In this chapter, we address the technical aspects of the approach used by Perspective Access Networks to overcome Internet fragmentation. Our central argument is that we can build an overlay network to bridge fragmented portions of the underlying network, and we show that a single set of interconnected *forwarders* can be used for this purpose. We presume that each part of the fragmented Internet is visible from at least one of these forwarders. There are three aspects to constructing the overlay network:

- Aspect "A": Construct a system for identifying perspectives such that clients can identify and describe the perspective from which they want to access Internet resources.

- Aspect "B": Construct a system for advertising perspectives through the network so that a path through the network from a client to a perspective can be determined.

- Aspect "C": Construct a system for transporting the application messages from the client to the application server once the path through the overlay network has been determined.

Chapter 4 addresses aspect "B." This chapter, which addresses aspects "A" and "C," is arranged into six sections. The first section gives an overview of the architecture and describe the principal architectural challenges. The second section describes some of the challenges to deployment. The third section carefully defines a language for describing perspectives and defines how requests for particular perspectives are to be formed. The fourth section provides details describing our prototype implementation, including requirements for the system that we use for the control plane. The fifth section describes how to extend PAN to incorporate authentication of clients by the forwarder offering the chosen perspective. The final section offers a detailed description of some practical uses of PAN.

Throughout the remainder of our discussion, we use the term *transport domain* to refer to a set of hosts S for which the network provides full transport services to all pairs $(a, b) \in S \times S$. In particular, for our purposes, S is a transport domain if and only if all pairs of hosts $(a, b) \in S \times S$ can mutually establish and maintain TCP sessions to each other.

3.1 Design Challenges

Next, we present the challenges associated with designing the protocol to be used by Perspective Access Networks. We consider the essential infrastructure components,

the forwarding of traffic through the network, privacy, identification of resources, separation of roles, and design of the control plane. These topics are covered, respectively, in the subsections of Section 3.1.

PAN itself consists of a peer-to-peer overlay network of *forwarders*, each of which has access to some set of Internet resources. Before a client can establish a connection to a forwarder, it must first have possession of a *descriptor* for that forwarder, which contains reachability information (for example, the IP address and TCP port of the service) and its public key. (The descriptors used by Tor (38) as described in Section 2.5.1 are sufficient for this purpose.)

To achieve universal access, we must provide a means by which all resources can be named. To this end, we stipulate that names of forwarders are globally unique within a PAN, and we identify some target resource R as a combination of the name of a forwarder that can reach R and the name of R as seen by that forwarder. Unlike Internet hosts, whose addresses are determined by location within the topology and whose names are apportioned by hierarchical DNS, a PAN forwarder *chooses its own name* by generating a self-certifying identifier (defined later) and using that as a global name. In this sense, each resource accessible via PAN is associated with at least one unique name, specifically the name resulting from the combination of the name of the forwarder and the name of the resource as seen from that forwarder. The novelty of this aspect of PAN is that it allows resources to be globally specified in the absence of hierarchy. However, the resources themselves are not responsible for guaranteeing global uniqueness—instead, all that is required is that *some* particular forwarder has the ability to identity the resource uniquely.

The PAN design does not require global agreement about apportionment of names in favor of allowing different regions of the Internet to have their own namespaces; a client can implicitly specify the relevant namespace by specifying the perspective from which it seeks to access a particular resource. This means that PAN allows us to relax the assumption that all names for Internet resources are globally unique.

The overlay network that connects all of the forwarders to each other consists of a *data plane* that carries tunnelled DNS requests and TCP sessions, as well as a *control plane* that carries routing information.

3.1.1 Infrastructure Components

The first design challenge involves determining the set of elements that compose Perspective Access Networks. PAN forwarders identify themselves in two ways, first by the self-certifying identifiers that they generate for themselves, and second by a set of characteristics that describe the perspectives and services that they offer. We describe the mechanical details of perspectives in Section 3.3.

Propagation of Perspectives

We seek to avoid requiring that clients query directories arbitrarily to learn about how to reach a perspective, so if a client uses a distributed directory service to find out more about some perspective with some set of characteristics S, then an entry describing how to reach a perspective matching S **should** exist in any particular directory that the client contacts. This means that routing information about S *should* propagate all the way from its source (see Figure 3.1). Clients must accept

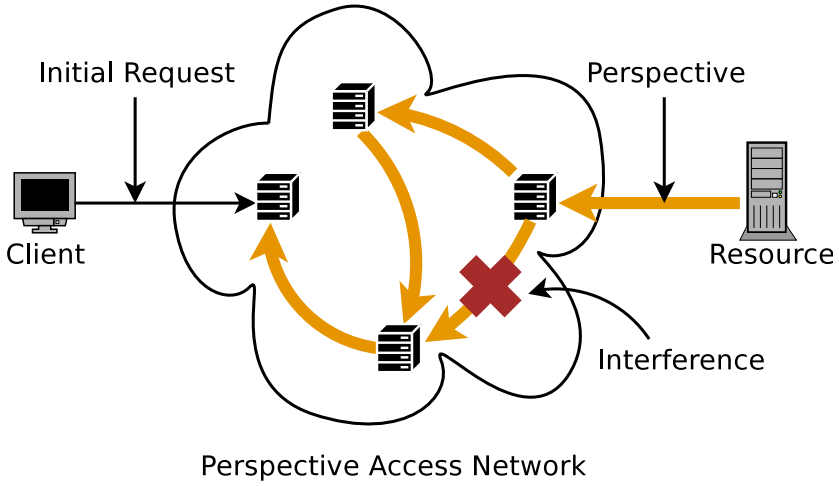


Figure 3.1: Propagation of Perspectives. *Forwarders propagate their perspectives through the network of directory servers so that they can tell clients how to reach their desired perspectives. (Perspectives may not propagate to all directory servers.)*

that information about a perspective might be filtered out and not propagated via the directory service (in which case they will either fail or fall back to querying), but generally, a directory will have the requisite knowledge to assist a client in routing its packets toward a perspective that suits the request.

Propagation of Forwarder Names

The self-certifying identifiers that specify individual forwarders are determined by applying a function to a value that the forwarders choose randomly. As a result, the names of two forwarders do not provide any indication of whether they are proximate to each other, and there is no way to aggregate forwarder names.

Therefore, if X is a PAN forwarder and we require that any user of the entire PAN network can identify this particular forwarder explicitly and uniquely by name, then the name X must propagate to all directory servers used by all clients wishing

to access a resource via X. This poses a significant challenge to scalability; for example, BGP routers can only handle a limited number of prefixes and still operate efficiently. For comparison, as of 23 January 2006, BGP routing tables contained a total of 179 thousand unique routing prefixes, and even if the maximum possible prefix aggregation is considered, then a BGP listener might expect to have 100 thousand unique entries in its table (118). In addition, the hardware present in BGP routers is designed with routing in mind; we expect that PAN forwarders will ordinarily run on general-purpose, commodity hardware. Similarly, PAN forwarders may lack the bandwidth available to BGP routers. So, for small communities with relatively few forwarders, the ability for directory servers to refer to each forwarder explicitly is feasible. In sufficiently large PAN networks, we expect that while a client might be able to refer to some subset of the forwarders explicitly by name, the client will be unable to refer to the vast majority of forwarders except by characteristic or membership in some sort of collection. We explore the scalability tradeoffs in detail in Section 5.5. Refer to Section 3.6 for some practical deployment scenarios for PAN.

PAN clients contact *directory servers* to obtain information necessary to route traffic to the forwarders of interest. Each directory server contains some routing information about perspectives. We do not assume that all directory servers share all routes with all other directory servers: if the operator of directory server D_1 does not want to share some perspectives with directory server D_2, then D_2 will not receive those perspectives from D_1. D_2 may also receive routes from another directory server that provides the perspective instead, just like in interdomain routing. An important difference is that while interconnection relationships between BGP autonomous

systems may involve some investment in physical infrastructure, interconnection relationships in PAN require only that peers can use the Internet to connect to each other. Nevertheless, it may be the case that the majority of the barrier to entry into such relationships is due to business decision process rather than infrastructure.

Directory updates in PAN are analogous to updates in BGP. Propagating a perspective is analogous to propagating a long IPv4 prefix: the advertiser provides the next hop for data to take *en route* to the destination. The path-vector algorithm propagates only one advertisement per forwarder per directory server, thus filtering undesirable routes and creating a tree rooted at the directory server to which the forwarder initially published. A client in any transport domain could use a local directory server to deduce an entire path in this fashion.

Since PAN provides access to perspectives in a meaningful way, we posit that the relationship between PAN and Virtual Private Networks is analogous to the relationship between the Internet and actual private networks. Extending the analogy, PAN forwarders are analogous to Internet routers, and PAN directory servers are analogous to BGP speaker-listeners.

While advertising entire routes for each forwarder to each directory server individually may be sufficient from the perspective of clients, it is also inefficient, since it would require all directory servers throughout the entire network to maintain path information for all forwarders. The process of propagating and maintaining consistent replicas of forwarder reachability information throughout the entire system could yield both excessively large tables as well as substantial network and processing overhead. We apply two techniques to address this problem: *semantically meaningful*

perspectives and *forwarder summaries*.

Semantically Meaningful Perspectives

A *perspective* is the view of the Internet that a particular forwarder provides. Rather than propagating individual forwarder information explicitly to its neighbors, directory servers may choose to propagate only *attribute sets*, which are semantically meaningful perspective metadata associated with individual forwarders or aggregates of such data. These metadata describe the salient characteristics of a perspective, such as location, policy, and functional capabilities. Metadata about individual perspectives are propagated in the same manner as forwarder information. Propagating perspectives rather than the names of forwarders carries two main advantages:

- Individual directory servers can implement *policies* that take advantage of these metadata to determine what kinds of perspectives should be propagated.

- Information stored at directory servers to describe what is available and how to reach it (henceforth we use the term *route* to refer to availability information for a single perspective) carries semantically meaningful information that is more useful to clients.

- Propagating routes for perspectives rather than for individual forwarders allows multiple forwarders with sufficiently similar perspectives to be grouped into a single category. Directory servers can store routing information for categories rather than the individual members, thus improving scalability.

Section 3.3 provides a detailed description of the several ways in which PAN represents individual perspectives. Propagation of metadata, including perspective data, is covered in Chapter 4.

Forwarder Summaries

Even if we stipulate that clients must be able to specify individual forwarders explicitly when building circuits, the PAN architecture allows directory servers to store and forward only next-hop reachability information for individual forwarders, so that a client seeking a particular forwarder will have enough information to determine and access another directory server along the propagation path to that forwarder.

Suppose that Alice is a forwarder who advertised her perspective to her local directory server, and Bob is a client who wants to be able to find Alice starting with a regular directory lookup. If PAN were arranged hierarchically, then we could use a DNS-like technique: either Bob or a directory server acting on behalf of Bob could ascend the tree and descend correspondingly to find Alice. However, PAN is not hierarchical, so the system must propagate information about the existence of Alice from her directory server to the directory server used by Bob. Refer to Chapter 4 for details.

Querying Directories

If Alice wants to talk to Bob from the perspective of Carol, then she will ask her local directory server for a means of reaching Carol. A successful response from the directory server will take one of two forms:

- The name of a directory that Alice can use to build a path to Carol, along with information for how to reach that directory.
- A list of forwarders through which data can be sent from Alice to Carol.

Details of the query protocol are covered in Chapter 4.

3.1.2 Forwarding Traffic

The second design challenge involves forwarding traffic and providing infrastructure to link perspectives together. In general, this means using a *proxy*, i.e., an intermediary willing to handle requests by forwarding traffic in both directions, to forward requests from one perspective to another. These requests in turn can be interpreted by the forwarder providing the perspective sought by the client.

For proxies at the network layer, we will need an encapsulation format that allows IP packets to be unwrapped and reconstructed at each forwarder. IP Address Encapsulation (IPAE) (32) provides a useful tool for implementing a new protocol close to the network layer while minimizing deployability concerns.

For proxies above the network layer, there are a number of well-known solutions. Perhaps foremost is the popular and versatile SOCKS protocol (78), a transport-layer proxy that provides a general framework for traversing firewalls. For our purposes, using SOCKS, it is possible to forward requests for any application protocol that uses TCP. Other popular proxies are application-specific, including HTTP proxies Squid[1]

[1]Squid, `http://www.squid-cache.org/`

and Privoxy[2]. Our prototype implementation, described in Section 3.4, uses both transport-layer and application-layer proxies.

Perhaps the most significant question surrounding our use of forwarders to provide access lies in the distribution of connection state. Even if we design the service to act below the transport layer, packets must be able to travel between the client and the resource in both the forward and reverse directions. There are several general ways of achieving this goal, including but not limited to the following:

- **Option 1.** Require that applications have knowledge of the specialized forwarding infrastructure. Applications would be able to manage the process of identifying and specifying resources, finding forwarders, and encapsulating datagrams to be forwarded. The provider of a resource would be responsible for directing replies back to the requester; perhaps the client application or intermediary forwarders could modify the application-layer header to provide hints that allow responses from the application server to propagate back to the client.

- **Option 2.** Build support for the forwarding infrastructure into the operating system of both the clients and the servers. As with TRIAD (23), use a protocol that encapsulates IP and contains a field for specifying forwarders between the source and the destination of a packet. Either have the original requester specify the set of forwarders explicitly, or allow this field to grow as each successive forwarder passes the datagram toward the application server. The application server or its operating system may use the list of forwarders to direct the response.

[2]Privoxy, a web-scrubbing HTTP proxy that can export traffic via SOCKS4A, `http://www.privoxy.org/`

- **Option 3.** Maintain connection state within the network. Each forwarder functions in a manner similar to a traditional NAT, maintaining a mapping for individual connections that allows them to correctly route replies. Transport-layer proxies allow this system to be implemented incrementally, without requiring substantial change on the part of clients, servers, or network infrastructure, but with the added costs associated with violating the end-to-end principle. Another point to consider is that maintaining state means that such state can be recovered. The core issue with maintaining state is node failure. Specifically, there are three potential exposure risks: (a) stored connection state may be suitable for subpoena in a way that ephemeral traffic is not; (b) Internet service providers may be given legal authority to collect stored connection state for their own purposes (cf. the 2004 Massachusetts case affirming the right of an ISP to monitor the email of its customers that was later overturned (135)); and (c) governments may require Internet service providers to keep records of traffic entering and leaving their networks (cf. European Union data retention directive (41)).

Since our objective is to provide perspectives from which current Internet resources can be accessed (and not to build an overlay network that happens to provide its own content or services), we choose Option 3, which entails maintaining connection state at forwarders within the network, even though this violates a central tenet of the end-to-end principle. We believe that the value of our network in circumventing existing technical barriers justifies our willingness to rely upon network elements to store connection-specific data.

Next, we must determine how much state we should maintain at individual forwarders and which forwarders should carry which aspects of the connection state. We must consider the implications of what happens when forwarders crash or lose state, not to mention whether individual forwarders might sometimes be forced to drop table entries to accommodate new connections, and the risk of a denial-of-service attack that exploits such an approach. Might it be possible to design a way of recovering connection state even if a forwarder along the path fails? Would there be a way to replicate state through the system in a manner that mitigates dependency upon one particular route through the set of forwarders for each connection? Industry researchers have devoted substantial effort to solving the problem of stateful failover techniques to provide redundancy to network address translators (25).

Our approach must also assure bidirectional communication between application clients and application servers. It is useful to look to anonymity systems for techniques, since such systems have an intrinsic need to address this problem: communicating with a party whose identity is hidden is similar to communicating with a party not reachable from the local perspective. One approach is to have the client explicitly provide a means by which the exit forwarder can route replies back through the infrastructure to the client. Mixminion (35) uses specialized *reply blocks* for this purpose. Other systems, like Tor (38) and ANON (76) explicitly establish circuits between a client and a forwarder. I3 (125) allows for the specification of *rendezvous points* that allow the client and server communicate indirectly. Architectures designed to accommodate indirection offer substantial benefits in achieving host mobility (148).

Ultimately, we believe that the most effective means of maintaining connection

state for individual requests is to build source-routed circuits. Our principal supporting arguments are as follows:

- Bidirectional Communication. By establishing circuits, we provide a return path by which datagrams received from the application server can be forwarded to the client. Forwarders do not need to know how to reach uniquely-identified clients, since the system effectively uses the same pipes for both forward and return traffic.

- Ordered Delivery. Most Internet traffic is TCP, so we will want the forwarder providing the perspective to send its messages in order if possible. This means that there is a high value for receiving datagrams in order from the client; having packets take a single path through the overlay facilitates this.

- Performance. Connection setup is expensive; either clients or forwarders must determine, through a series of lookups, how to forward datagrams through the overlay. By constructing circuits, we allow clients to bear this burden as a one-time cost; once the circuit is built it can be reused for the remainder of a potentially long session. Also, the public-key cryptographic handshakes necessary to authenticate forwarders along the path are expensive relative to the symmetric-key operations needed to carry data; by building circuits, we can establish a session key once for a circuit that can be reused over and over. (An important disadvantage of the circuit-building approach is susceptibility to denial-of-service attacks that take place after an application has committed to using a particular circuit).

- SECURITY. Source routing allows a client to specify the entire path through the overlay to the exit forwarder, and onion routing using a system like Tor allows clients to verify each hop along the path that individual datagrams take. This verification is particularly important when a client wants to ensure that the system is providing a perspective that meets the specified criteria. (Note that it would be possible, though perhaps more cumbersome, to use onion routing in a network-layer forwarding system like ANON (76), yielding similar security advantages.)

3.1.3 Privacy

The third design challenge is determining what information to expose, both to observers and within the network. Unlike some distributed proxy networks, PAN does **not** intend to provide anonymity, though it may be used to address the natural conflict between anonymity and the unpredictability that occurs from choosing randomly from a set of different perspectives. Since our system provides connectivity between end hosts in a manner that may prove incompatible with the interests of operators providing service between the end hosts, we must choose whether the overlay network of forwarders is to be *secret* or *public*. In a *secret* network, the identities of participating forwarders are deliberately obscured, so that while a participant in the system must know the identities of a small number (possibly one) of other participants, it has no way of ascertaining the identities of other forwarders. Also, eavesdroppers have no means of determining whether a given host is participating in the network. In a *public* network, the identities of participating forwarders are exposed to public view, such

that any party may determine the identities of all of the participating forwarders, or whether any given host is a participant in the network.

One might argue that it is difficult or even impossible to create overlay networks that are truly secret. There are two essential reasons for this. First, encoding network traffic in a manner that looks like other traffic is difficult. This is essentially the steganography problem, which research has demonstrated to be an arms race (8). It would be possible to encrypt the traffic to look like SSL connections, for example, but a disproportionate quantity of SSL connections on any given link would arouse reasonable suspicion. Second, even if it were possible for a system to encode its traffic such that it is indistinguishable from other traffic on the link, numerous timing attacks and attacks on various links in the system could be used to determine the identities of the forwarders.

Furthermore, if forwarders can know each other's identities, they may be able to optimize communication through the overlay in ways that would not be feasible if they were denied access to each other's identities. The authors of Tor provide some good arguments for why a public system is both more economical and more practical than a secret system (38), and we intend for our overlay network to be public as well. There are several consequences of this decision; in particular, consider the case in which one user has an upstream Internet service provider who wants to filter communication between that user and some particular end host.

Like overlay networks that provide anonymity, overlay networks that provide users with the ability to select their perspectives take a clear position in the tussle between users demanding greater liberties and the governments and service providers who seek

to limit or otherwise constrain their activity (28). A secret system aspires to allow end hosts to communicate even in the event that powerful intermediaries intentionally deny direct access, and a public system makes its participants known, allowing adversaries who control network infrastructure to deny access to the entire overlay network as a whole (we will return to this point in Chapter 5).

One solution to this problem is "multiplexing" traffic that an adversary has an interest in filtering with traffic that an adversary has an interest in *not* filtering. Suppose that there are two kinds of resources: resources to which the intermediary is for some reason compelled to provide access, and resources that the intermediary would prefer to filter. While it may be possible to combine the two in an encrypted channel, the benefits offered by obfuscation of this sort may be difficult to measure. Moreover, ease of filtering is important to those offering online services, providing them with a way of blocking access originating from the overlay network if they so choose. Finally, a public system stands a better chance of being embraced by Internet service providers.

Generally speaking, choosing to design a public system means not being able to provide guaranteed service through networks whose operators are interested in deliberately filtering content. Thus, network operators like the government of China could easily configure border routers surrounding networks in China to disallow contact with known forwarders in our overlay, and an enterprise interested in controlling the set of hosts with which its internal users may communicate could use a similar technique. For these reasons, we consider the space of the problem that we intend to address to include the following areas: BGP misconfiguration, network malfunction, policy

decisions of network operators controlling areas of the network on the critical path between two hosts who want to communicate, policy decisions based upon convenience or incomplete information, and policy induced by the use of in-band mechanisms to address access-control problems (refer to Section 1.1.2 or examples of these forms of fragmentation). We do not intend to address adversaries who control critical path infrastructure between forwarders, or between users and forwarders, and who intend to explicitly restrict the use of the forwarding infrastructure. (Nevertheless, we do consider some adversary models that involve weak adversaries who lack the means or the intent to efficiently block all nodes in the overlay; refer to Section 3.6 for details.)

3.1.4 Identification of Resources

The fourth design challenge is determining a method for identifying resources accessible via a Perspective Access Network. The decisions in this space are vital to defining the scope of compatible implementations, since they define the method by which users can specify resources outside their local purview.

We first consider the abstract problem of how to refer to resources. The current Internet refers to resources by either (a) IP addresses unique to the network, which provide a description of the location of a resource and a local identifier, or (b) DNS names, whose allocation is the subject of much contention (141). An alternative might be to use a system like *intentional naming* (3), in which each resource is known to applications by some descriptive name (which may be a function of its characteristics, for example) rather than its location within the network. In Perspective Access Network, we use a combination of both: by using semantically meaningful descriptions

of perspectives, we extend the meaning of "location" to potentially make it more relevant to businesses and end users, while still preserving some notion of referring to resources by location.

Second, we assert that the names of forwarders must be unique within a PAN. By requiring global uniqueness, we have assurance that two requesters who refer to the same network location as viewed from a forwarder with a given name are interested in the same service. Enforcing global uniqueness is generally difficult, and part of our goal is to avoid the requirement of a central authority. However, there are ways of achieving an approximation of global uniqueness (within a PAN) even if strict global uniqueness (within a PAN) is impossible. One way is for each peer to choose a random integer from a certain interval. If the interval is sufficiently large, then we can be assured that there are no conflicts with high probability. One way to provide some guarantee of uniqueness is to use *self-certifying keys* (52; 105), a method of implicitly verifying the holder of a key by using the key. Self-certification allows us to embed into the identifier of a resource a string that specifies the public key corresponding to a private key known only by the peer in possession of that resource. The Self-Certifying File System (83) applies this technique to certify path names, and the technique may be used to allow providers of resources to authenticate themselves without the need for a central authority.

Use of public-key technology to provide *long-term* authenticated identities is not strictly necessary. After all, most authentication of services in the modern Internet, virtual private networks and IPSec (73) notwithstanding, is performed at the application layer. In this case we may still want to establish a means by which individual

peers can know that they are communicating with the same party with whom they had communicated at some earlier time. For this purpose, we consider the use of Purpose-Built Keys (PBK) (15), which allow the initiator of a conversation to prove its identity in the future. PBK works by having the initiator generate a public/private key pair during its first conversation with the receiver, associates it with an ID, and sends both the ID and the public key to the receiver. Subsequently, when the initiator needs to prove its identity to the receiver, it sends the ID to the receiver and receives a challenge, by which it demonstrates knowledge of the associated private key. We envision the possibility of using a combination of short-term self-certifying identities and PBK to authenticate forwarders to clients, though our prototype implementation does not use this approach.

Perhaps the most important problem of naming is the question of how resources describe themselves to an eager public. If we assume that the Internet is not fragmented, and that everyone can access everything made available to the "public" Internet core, then DNS names are sufficient to describe resources. However, as the Internet becomes fragmented, naming becomes more difficult. While a DNS name is still useful if the perspective is known or assumed, providers of Internet resources in a fragmented world need a means of characterizing the set of perspectives that would be sufficient to allow access to the resources.

A related problem is that of how to identify connection endpoints, given that clients may not be able to determine whether two resources are the same given that they are viewed from two different perspectives. The unique, long-term, self-generated, self-certifying endpoint identifiers proposed by Balakrishnan et al. (6) may

address this problem. The details are developed further in the design of DOA (142).

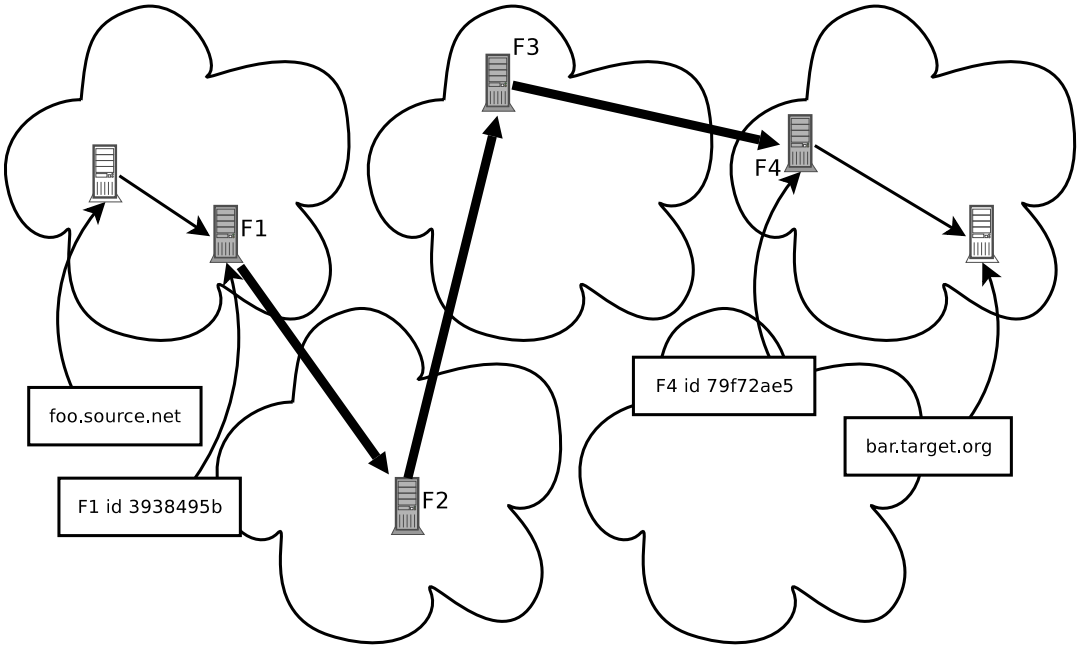


Figure 3.2: Accessing a Resource. *The source establishes a connection to* `bar.target.org` *from the perspective of* F_4. *DNS requests and TCP sessions are both tunneled through the infrastructure.*

3.1.5 Separation of Roles

The fifth design challenge lies in determining how the system should be organized and which roles are played by the individual components. Suppose that the forwarders have organized themselves into an overlay that can forward TCP traffic. Each forwarder independently generates a self-certifying identifier, and forwarders throughout the system refer to other forwarders using these identifiers (see Figure 3.2). As long as the size of the identifier is sufficiently large and the sources of randomness are sufficiently effective, the chance of a namespace collision among these identifiers within

the system will be negligible.

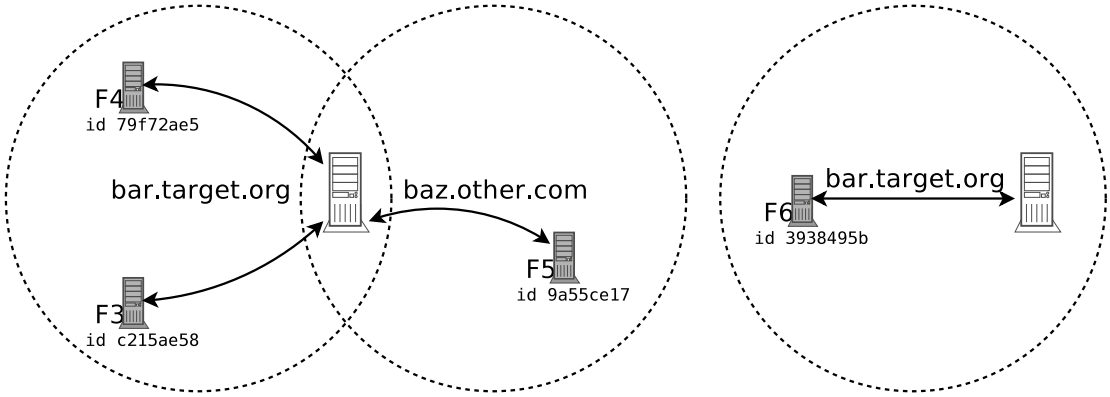


Figure 3.3: Multiple Names. *A resource need not have only one DNS name within a PAN. In this example, the target host is known to F_3 and F_4 as* `bar.target.org` *and to F_5 as* `baz.other.com`*. Meanwhile,* `bar.target.org` *from the perspective of F_6 describes an entirely different resource.*

Observe that the combined name "`bar.target.org` as seen from F_4" is globally unique, but the name was not apportioned by any authority of global scope. Also, there is no requirement that each resource is associated with exactly one forwarder; multiple forwarders may be able to reach the same resource, possibly using different names. See Figure 3.3. In the example provided, the source host may use any of "`bar.target.org` as seen from F_3," "`bar.target.org` as seen from F_4," or "`baz.other.com` as seen from F_5" to refer to the target server. Another useful feature of this design is that if the network-layer address or name of a forwarder changes, its PAN name does not, thus promoting mobility of forwarders.

3.1.6 Control Plane

The final design challenge is designing the control plane. PAN requires clients to build source-routed circuits to the exit forwarder, which provides the requested

perspective. Thus, the objective of routing in PAN is to provide clients with a means of learning how to build a source route to the exit forwarder specified by its PAN name. There are two fundamental approaches to executing this task, *advertisement* and *querying*.

In the *advertisement* approach, each forwarder announces its availability to the entire overlay network using a path-vector flooding protocol similar to BGP. The primary constraints to this method are that directory servers must be adequately apprised of changes to availability, and each directory server must maintain entries that collectively describe all of the currently available forwarders in the system. Scalability of the network is a serious concern with this approach, since (a) all hosts need to know about all perspectives, (b) self-certifying identifiers cannot be aggregated in any meaningful way, and (c) even semantically significant perspective metadata may be difficult to aggregate.

An alternative approach is *querying*, in which a client who wants to find a route to some perspective issues a request that is propagated through the network. To reduce the cost associated with this operation, we may stipulate that each directory server should maintain cached entries corresponding to recent queries. Scalability is a concern for this approach as well, but in a different way: as the network grows, the process of querying takes longer, and many low-latency Internet applications such as web browsing require short connection setup time. Generally, we seek to minimize connection setup time, so that users can establish new circuits in real-time.

We believe that a combination of *advertisement* and *querying* provides the best results: aggregation strategies can alleviate some of the burden associated with prop-

agating advertisements throughout the network, and targeted querying can be efficient, provided that clients have enough information to know which directory servers to query. In Chapter 4, we describe the roles of advertisement and querying in the PAN directory service.

Another challenge lies in associating meaningful attributes with individual forwarders. For example, a user of the system may want a means of accessing a resource via a forwarder in the Netherlands, but it need not matter which forwarder specifically. Similarly, a user may want a means of accessing a resource via any forwarder that provides access to politically-themed blogs. In Section 3.3, we describe a means by which a requester can specify a set of attributes rather than a particular destination forwarder.

3.2 Deployment Challenges

Now that the architecture is in place, we focus upon the principal *technical problems related to PAN deployment*, which include a number of questions about service discovery, network organization, and the overall usefulness of the system. Foremost are the concerns involving how clients learn of the existence or availability of resources and how forwarders know how to connect themselves to the network and advertise themselves properly. This section addresses the following questions:

- *How does a forwarder discover the resources that it can access?*
- *How does a forwarder discover other forwarders?*
- *How does a client discover forwarders?*

- *How does the system ensure that forwarders can mutually reach each other?*
- *How does the system handle namespace collisions?*

3.2.1 Resource Discovery

How does a forwarder discover the resources that it can access? This is essentially a question of *self-identification*: should we require that forwarders learn about their environment prior to advertising their existence to the network? If we assume this requirement, then we can envision several solution candidates:

- AUTOCONFIGURATION. A forwarder learns about its perspective by observing the network-layer configuration of the operating system upon which it is running. For example, it might learn about the address range of its local network, whether that range is public or private, whether there are multiple network interfaces, the physical layer media on each interface, etc.
- ACTIVE PROBING. A forwarder could learn about its perspective by actively scanning its environment. For example, it could launch random port scans to determine what sites it can reach and what ports are unfiltered. At a higher layer, it might contact individual application servers to determine either (a) whether the content provided by a particular application server matches what it expects to find, or (b) whether the application server presents a valid certificate.
- SERVICE REGISTRATION. A forwarder could learn specifically about various services to which it has access using a registration process. This might mean either (a) stipulating that providers of services explicitly register the services

with the forwarder in a manner similar to that employed by I3 (125), or (b) stipulating that forwarders use some underlying autoconfiguration protocol such as Universal Plug-and-Play (134) or Zeroconf (144; 61) to discover resources within its local area. Specifically, there are several existing protocols designed with network browsing in mind, including DNS-Based Service Discovery (24).

On the other hand, perhaps having forwarders actively discover resources within their local area is not strictly necessary; forwarders could certainly be entirely passive instead. Consider, for example, the possibility of on-demand discovery via DNS lookups. A client could request a resource from a forwarder by first sending an ordinary DNS request. The forwarder would then issue the request on behalf of the client, and if there is a negative response or no response, then it knows that it is unable to fulfill the request. Of course, this particular approach means additional delay for the client, though clients could sometimes benefit if results are cached within the directory.

A far simpler solution might be to require people deploying forwarders to configure them with appropriate reachability information, but this might sometimes be overly burdensome from a usability standpoint. For now, we require only that the directory service have some uniform description of the perspective provided by an individual forwarder. While we do not specify explicitly how that description is obtained, we provide a means by which forwarders can self-identify by providing metadata to the directory service.

One of the salient features of PAN is that it provides access to existing, legacy resources, i.e., resources agnostic of PAN itself. The PAN architecture does not

provide a means of tracking these resources. For example, while it may be possible to determine that certain URLs yield substantively different websites when viewed from Germany than when viewed from the US, trying to track all such discrepancies would be little short of impossible. That said, it would certainly be possible to use PAN to discover and catalogue such discrepancies.

3.2.2 Network Arrangement

How does a forwarder discover other forwarders? This is the fundamental question of how forwarders organize. We might imagine an infrastructure in which forwarders are completely incognizant of each other. Requests to forwarders may be encapsulated in other requests to forwarders in a manner similar to onion routing, and clients could describe resources not by particular perspectives, but by explicit chains of forwarders that happen to lead to the desired perspectives. The problem with this approach is that requesters of services are then faced with the responsibility of performing all network discovery, a task that undermines all of the benefits of a routing infrastructure.

Alternatively, we might implement all forwarders as equal peers, all configured to function as directory servers that receive the global routing table, all sending and receiving network reachability updates that affect the global routing table. This is impractical for several reasons. First, we imagine that most forwarders in a functional PAN will be *leaves*, meaning that while they will act as exit forwarders to provide access to resources, they will **not** route traffic to other forwarders. The argument for why leaves should not participate in the global routing table is analogous to the

interdomain routing argument for why single-homed autonomous systems should not use public autonomous system numbers: the additional overhead is both costly and superfluous.[3] Second, we can exploit hierarchy by having some forwarders act as directory servers to speak for large sets of forwarders that share some characteristic such as physical proximity or access to specific kinds of services. Finally, different forwarders will have different network connectivity. Some will have Internet connections that make them suitable as forwarders or directory servers, while others will not.

Therefore, we stipulate that a forwarder that participates in this network need not perform all functions. Individual forwarders should not be required to forward traffic to other forwarders, nor should they be required to participate in the directory service. Similarly, we do not require forwarders to provide perspectives to clients: for example, individual forwarders may be configured to participate in the global directory and to route datagrams to other forwarders but not to send data to application servers external to the PAN. By decoupling the various functions, we allow greater flexibility for deployment.

At the same time, directory servers tell clients to which directory servers to extend their circuits next, and directory servers need to be in-band to correctly detect service interruption. So, just as BGP speaker-listeners are also routers, all directory servers are forwarders, though some (probably most) forwarders are not directory servers. Forwarders that are not directory servers publish their perspective information to directory servers; each forwarder may publish to any number of directory servers.

Another paradigm that uses inequality to improve the scalability of overlay net-

[3] ARIN (`http://www.arin.net/`) no longer assigns autonomous system numbers to autonomous systems that are not multi-homed.

works is *landmark routing*, which takes advantage of knowledge of the underlying network. Brocade (147) uses this technique to dynamically form "supernodes" that provide substantial performance benefits; perhaps a similar technique could be applied to our system as an optimization.

3.2.3 Forwarder Discovery

How does a client discover forwarders? Ultimately, to discover forwarders, a PAN client **must** know the address of at least one directory server. If a single directory server provides access to a PAN that provides access to all resources to which a client requires access, then that directory server alone is sufficient. We choose to deploy a set of well-known directory servers for this purpose; however, we do not require any sort of central administrative structure for the directory servers. For clients in regions of the network without direct access to any of the directory servers, we stipulate that they must find reachable directory servers by some other means. Perhaps the network location of directory servers can be distributed out-of-band, possibly via DNS.

3.2.4 Unidirectional Links

How does the system ensure that forwarders can mutually reach each other? In some cases, Internet fragmentation occurs because two hosts cannot communicate with each other: forwarder Alice cannot talk to forwarder Bob, and forwarder Bob cannot talk to forwarder Alice. The solution that we have described thus far provides a means by which they can talk with each other, provided that forwarder Alice and forwarder Bob can each initiate conversations with each other via the same

PAN. However, this presumes bidirectional communication, and several forms of Internet fragmentation are the result of links that can be considered *unidirectional*: new conversations can be initiated from one side only. In particular, forwarder Alice may be able to initiate a connection to forwarder Bob, but not vice-versa. Both network address translators and firewalls that block inbound TCP connections to protected networks create unidirectional links. In both cases, forwarder Alice from our example is on the side of the private network, and Bob is somewhere on the outside.

We presume that for every interface between transport domains A and B, one forwarder T_A exists in transport domain A, one forwarder T_B exists in transport domain B, and, without loss of generality, T_A can open a conversation with T_B. If we want to build a "bridge" between transport domains A and B, then T_A must open a *persistent connection* to T_B. A detailed description of this process along with illustrative diagrams are provided in Section 3.4.3. We define any tunnel from X to Y as a *persistent connection* from X to Y if it meets the following criteria:

- X and Y can communicate freely and bidirectionally through the tunnel.
- If X notices that the connection has been severed for whatever reason, X establishes a new connection to Y, creating a new tunnel.

Given the preponderance of unidirectional links in the Internet today, the ability for individual forwarders to establish persistent connections is vital to PAN.

3.2.5 Namespace Collisions

How does the system handle namespace collisions? Since PAN allows

different resources with the same name to exist within the context of different perspectives, we are left with a number of questions about how the world will respond to the resulting namespace collisions.

If we assume that all names are universal, then no two organizations can choose to use the same name to identify their respective services, even if such services exist within different localities and do not compete. One result is that the organization that fails to procure the name may be accused of failing to preserve its own trademark. Burgeoning litigation surrounding domain name disputes has become increasingly expensive, leading to the implementation of the ICANN Uniform Dispute Resolution Policy[4] (ICANN UDRP) (64). One solution to this problem might be to have one organization reserve the name and transparently redirect clients based upon their geographic location to services provided by another organization. However, generally speaking there is no way to guarantee cooperation among organizations.

Since PAN architecture provides a means of avoiding the constraints associated with universal names, we are able to provide a workaround that achieves some of the goals sought by semantic-free references (141), though competition within individual perspective spaces may continue to exist.

3.3 Managing Perspectives

In a Perspective Access Network, forwarders offer a set of characteristics, or *perspectives*, that allow clients to specify from which location they want to view the Internet (or other networks). Next, we address the problem of describing perspec-

[4]ICANN Domain Name Dispute Resolution Policies, `http://www.icann.org/udrp/`

tives (design aspect "A" identified at the beginning of this chapter).

When PAN directory servers are informed of the availability of certain perspectives, they propagate the perspectives through the network according to their locally-configured policies. A client requests a perspective matching a set of characteristics, and receives instructions for constructing a source route through the network that provides a perspective meeting the specification.

This section defines the characteristics inherent to individual perspectives by identifying the essential ways in which network locations differ. In Chapter 4, we argue in favor of a flexible design for propagating perspective information through the network, with consideration for both local policies and the tradeoffs inherent to balancing query latency, query expressivity, and scalability goals. Using RPSL (4), the industry-standard policy description language used for BGP, we create new `route-set` and `filter-set` classes to accommodate our scheme for representing perspectives.

3.3.1 Defining Perspectives

Defining the set of perspectives available to clients first requires an assessment of which aspects of network location information are most salient. We specify six methods by which individual perspectives can be described.

- 1. Political Location. *(hierarchical)* This field provides the location of a perspective in terms of political jurisdictions. (e.g., `US.Massachusetts.-Cambridge`)
- 2. Network Name. *(hierarchical)* This field provides the location of a perspective in terms of organizational boundaries. This field is useful for describing

private networks (including ISPs). (e.g., `Harvard.EECS`)

- 3. Implicit Environmental Filtering. *(categories)* This field provides the names of broad categories of content filtering, with the goal of characterizing filtering policies by references to the nature of what is filtered rather than its effects (such as threats to open or democratic society). One example might be to include well-known names for the categories described in previous ONI reports, e.g., "News Outlets," "Sex," "Blogs," "Hate Speech," "Government," etc. (100). Another example might be to assume well-known names for filtering performed by certain organizations, e.g., "China," "Saudi Arabia", which could be shorthand for some set of more specific characteristics. So, this field contains a list of categories, each prefixed with a `'+'` character, meaning that the environment "accepts" this category, or a `'-'` character, meaning that the environment "rejects" this category. All characteristics are considered `'+'` by default, indicating no filtering. (e.g., `"pro-democracy"`)

- 4. Explicit Environmental Filtering. *(address ranges)* This field specifies particular network address ranges to which the perspective allows or restricts access. As an example, we use a list of CIDR prefixes, combined using `'+'` and `'-'` notation as described above. (e.g., `212.58.226.0/25`)

- 5. Geolocation. *(latitude-longitude coordinates)* This field provides the geographical (latitude and longitude) coordinates of the perspective to some degree of accuracy. We use degrees of arc to measure both the coordinates and the degree of accuracy. The field is a triplet consisting of (1) north latitude, (2)

east longitude, and (3) accuracy. Negative coordinates refer to southern and western hemispheres, respectively. (e.g., `42.3, -72.1, r:3km`)

- 6. Functional Capabilities. *(categories)* This field is a set of special attributes succinctly describing the functional advantages and disadvantages of this network location. For example, this field may include an indication of whether the perspective is behind a NAT, whether voice-over-IP data traffic patterns are allowed, and other policies and functional features specific to the network in which the perspective is situated. (e.g., `"no-long-term-connections"`)

field name	*format*
Geolocation	`FORMAT = ORD "," ORD "," ORD` `ORD = ["-"] *DIGIT "." *DIGIT`
Political Location	`FORMAT = *ALPHANUM *("." (*ALPHANUM / "*"))` `ALPHANUM = (DIGIT / ALPHA)`
Network Name	`FORMAT = *ALPHANUM *("." (*ALPHANUM / "*"))` `ALPHANUM = (DIGIT / ALPHA)`

Environmental Filtering (explicit)	`FORMAT = RULE *("," RULE)`
	`RULE = ("+" / "-") ADDR "/" 1*2DIGIT`
	`ADDR = 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT`
	`"." 1*3DIGIT`

Environmental Filtering (implicit)	`FORMAT = ("+" / "-") *ALPHANUM`
	`*("," ("+" / "-") *ALPHANUM)`
	`ALPHANUM = (DIGIT / ALPHA)`

Functional Capability	`FORMAT = ("+" / "-") *ALPHANUM`
	`*("," ("+" / "-") *ALPHANUM)`
	`ALPHANUM = (DIGIT / ALPHA)`

Table 3.1: Route-Set Field Formats.

Just as *prefixes* describe individual routes in the context of BGP routing, *perspectives* describe individual forwarders in the context of PAN. Our policy language

extends the RPSL `route-set` class to include perspectives; each perspective contains a set of fields consisting of either zero or one of each of the six fields identified above. Table 3.1 provides the ABNF format for each field.

Individual forwarders propagate the information about themselves as metadata to the directory servers, which in turn propagate these metadata (optionally with aggregation) to other directory servers according to locally-configured policy. Section 3.3.2 describes the mechanism by which clients may formulate queries using metadata from some subset of the aforementioned categories. The directory servers interpret queries as well as they can and respond to clients appropriately.

3.3.2 Selecting Perspectives

Clients select perspectives by issuing *metadata queries*, which match data from the perspective fields described above. For each perspective, an individual metadata query returns either **true** or **false** depending upon whether the perspective matches the query. Metadata queries take the following forms:

- 1. Political Location. *(Query format: prefix)* The query returns true if and only if the prefix specified is a prefix of the perspective, e.g., a query `a.b.c` would match `a.b.c.d` but not `a.b`.
- 2. Network Name. *(Query format: prefix)* Same as for Political Location.
- 3. Implicit Environmental Filtering. *(Query format: `+/-` bit, category name)* The query returns **true** if and only if the category name is listed (either implicitly or explicitly) as accepted by the perspective (if `'+'` is specified) or

not accepted (if `'-'` is specified).

- 4. Explicit Environmental Filtering. *(Query format: address)* The query returns **true** if and only if the query address is included within the set of addresses accepted by the perspective.

- 5. Geolocation. *(Query format: modal-operator, position, and range.)* The query returns **true** if and only if a given perspective is (**may** or **must** be, as determined by whether *possibly* or *necessarily* is indicated) within the number of degrees of arc specified by *range* of the coordinates specified by *position*. The difference between *possibly* and *necessarily* is that *possibly* is "liberal," allowing the inclusion of any perspective that is within the degrees specified by *range* **plus** its indicated error, whereas *necessarily* is "conservative," requiring a perspective to be within the degrees specified by *range* **minus** its indicated error.

- 6. Functional Capabilities. *(Query format: `+/-` bit, category name)* Same as for Implicit Environmental Filtering.

Metadata queries are used to filter perspectives. Our policy language extends the RPSL `filter-set` class to include these queries; each perspective contains a set of fields consisting of some number (possibly zero) of each of the six query types identified above. Table 3.2 provides the ABNF format for each query type.

query type	*format*

Geolocation	`FORMAT = "geo:" ("possibly" / "necessarily")`
	`":" ORD "," ORD "," ORD`
	`ORD = ["-"] *DIGIT "." *DIGIT`

Political Location	`FORMAT = "loc:" *ALPHANUM`
	`*("." (*ALPHANUM / "*"))`
	`ALPHANUM = (DIGIT / ALPHA)`

Network Name	`FORMAT = "net:" *ALPHANUM`
	`*("." (*ALPHANUM / "*"))`
	`ALPHANUM = (DIGIT / ALPHA)`

Environmental Filtering (explicit)	`FORMAT = "eef:" ADDR`
	`ADDR = 1*3DIGIT "." 1*3DIGIT "." 1*3DIGIT`
	`"." 1*3DIGIT`

Environmental Filtering (implicit)	`FORMAT = "ief:" ("+" / "-") *ALPHANUM` `ALPHANUM = (DIGIT / ALPHA)`
Functional Capability	`FORMAT = "cap:" ("+" / "-") *ALPHANUM` `ALPHANUM = (DIGIT / ALPHA)`

Table 3.2: Filter-Set Field Formats.

3.4 Implementation (Blossom)

In this section, we describe the special characteristics of Blossom, our prototype implementation of PAN. Blossom makes use of the onion-routing system Tor (38) for constructing circuits and transporting data. However, Blossom uses an alternate network discovery algorithm and its own directory servers. Unlike Tor directory servers, Blossom directory servers construct routing tables, using a path-vector protocol with an expressive policy framework.

The most interesting aspect of Blossom is how it interacts with other systems in the real world. This section describes those interactions, and through our description of the interaction between Blossom and Tor, we address design aspect "C" identified at the beginning of this chapter. Since Blossom directory servers generally do not

interact with systems outside Blossom, we were free to implement them according to the directory service that we describe in the next chapter. The current status of the Blossom directory service is that peering directives are fully implemented, metadata queries for perspectives are partially implemented, and policy directives are unimplemented. (Thus, we were able to conduct tests and demonstrate Blossom as a proof-of-concept implementation, but we continue to work with the Tor developers to improve its usefulness to the general public.)

3.4.1 Transport Layer Requirements

It is possible to implement Perspective Access Networks using a variety of systems that provide transport for client datagrams; however, not all data planes are created equal. PAN has a number of specific desiderata for its transport layer; we list the more important requirements below. It turns out that Tor provides a convenient controller interface that satisfies most of the requirements; the interface is the most significant factor in our decision to choose Tor for the substrate of our prototype implementation.

The following capabilities of Tor make it particularly suitable as a substrate for Blossom:

- Access Existing Internet Resources. Tor provides generic forwarders that are capable of acting as a proxy to access Internet resources agnostic of PAN on behalf of clients.

- Require No Special OS Configuration. Tor runs without special privileges, kernel hacking, or OS-level configuration, and it must be portable to all

sufficiently popular operating systems.

- Interpret SOCKS Requests. Tor affords Blossom the ability to interpret SOCKS requests (hostname and port information) directly. Among other things, this capability provides the flexibility to allow users and applications to embed perspective-specific instructions in the hostname field.

- Direct the Construction of Circuits. Given a set of descriptors and instructions to build a circuit, Tor is able to establish a secure, authenticated tunnel. In general, a substrate for PAN must afford PAN clients the ability to (a) provide their own descriptors, (b) define the circuits to be constructed, (c) know when circuits succeed and fail, and (d) open multiple circuits simultaneously. The substrate **should** also allow a PAN client to extend or cut the length of a circuit on demand.

- Attach TCP Streams to Circuits. The Tor Control Protocol provides a means by which PAN can (a) know when a TCP stream has become ready and (b) allow PAN to attach TCP streams to specified circuits on demand. In general, a PAN substrate **should not** close circuits built by the PAN client until authorized. The substrate **should** provide a means by which the PAN client (a) knows when a stream closes, (b) knows when a stream attaches successfully, and (c) can change the circuit to which a stream that has not yet been successfully attached is assigned.

- Establish Persistent Connections. Tor provides a means by which a PAN forwarder can establish persistent connections to other PAN forwarders of

its choosing. This capability is essential in allowing PAN forwarders to construct bidirectional tunnels.

- MANAGE CIRCUIT EXTENSIONS.[5] It is possible to modify Tor to allow PAN forwarders to mediate the construction of circuits as they choose. This capability allows PAN forwarders to implement policy that limits or forbids the extension of circuits from one particular neighbor to another.

3.4.2 Integrating Blossom and Tor

Both Blossom networks and Tor networks consist of interconnected proxies, but where Tor chooses to optimize for anonymity, Blossom chooses to optimize for perspective-specific reachability instead. Tor achieves its anonymity goals by having clients build source-routed circuits at random using the deployed network of roughly 600 servers[6] around the world: the choice of *exit forwarder*, i.e., the last hop of a circuit and the perspective from which clients view the Internet, is left to chance. If the chosen exit forwarder happens to be in Germany, then Google search results will be skewed to assign German-language pages a higher rank by default. If the chosen exit forwarder happens to be in China, then clients will not be able to access all of BBC News. These discrepancies may appear small, but ultimately, the Internet has no neutral locations; both the network and application servers may ascribe different semantic meanings to different network locations. Different locations mean different access, and Blossom provides a system for selecting, in a uniform way, which location to use.

[5] Tor does not yet have this functionality as of version 0.1.1.20, June 2006.

[6] as of June 2006

Blossom achieves this goal by sacrificing many of the stronger anonymity benefits of Tor: indeed, a user's anonymity is degraded if her choice of circuit reveals information about her preferences or interests. But, choice has value in itself, and Tor provides a useful general-purpose substrate upon which higher-level services may be built, even services that do not include anonymity as a goal. For an overview of how Tor works, refer to Section 2.5.1.

Blossom networks have other advantages over the Tor network as well. For example, the Tor network assumes that all forwarders are mutually reachable, while the Blossom network makes no such assumption. In fact, the Blossom overlay network supports arbitrary topologies.

Blossom uses Tor descriptors, since they are necessary to build circuits using the underlying Tor system. Like ordinary Tor forwarders, each Blossom forwarder pushes its Tor descriptor to directory servers. However, Blossom directories carry some additional reachability information that make routing possible by describing the network topology, in which possibly not all forwarders can directly reach all other forwarders. We describe the Blossom directory service in Chapter 4.

A single client can be used to access both Blossom and Tor networks, though the client will need to know about both Tor and Blossom directory servers. Blossom directory servers are integrated into the Perspective Access Network, and the Blossom client itself uses the Tor Control Protocol[7] to exchange information with Tor and issue instructions. Also, since Blossom encodes perspective requests in the hostnames that are sent to the SOCKS proxy, use of Blossom may require an additional application-

[7]TC: A Tor Control Protocol, Version 1, `http://tor.eff.org/cvs/tor/doc/control-spec.txt`

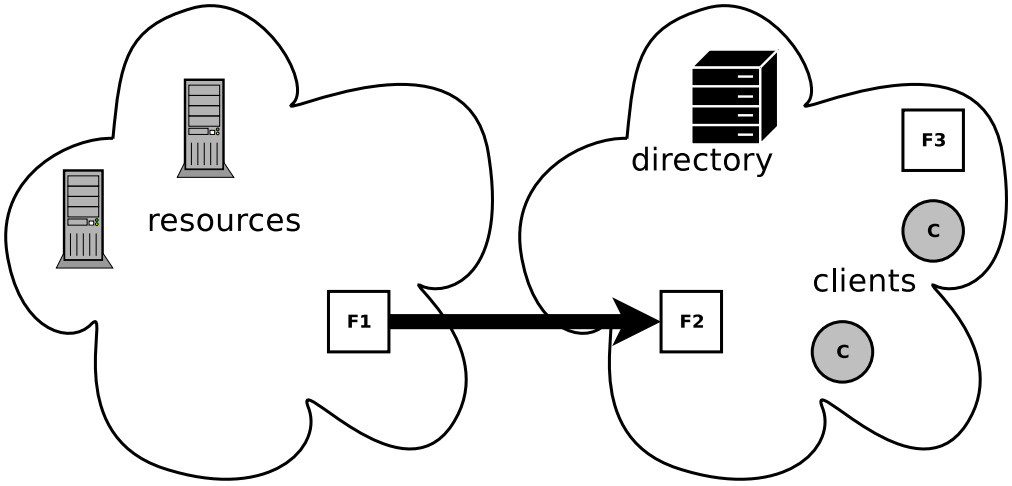


Figure 3.4: ESTABLISH PERSISTENT CONNECTION. *If F_1 wants to provide access to resources otherwise not accessible to clients in the vicinity of F_2, and if clients in the vicinity of F_2 cannot reach F_1 directly (as shown by the black unidirectional arrow), then F_1 must first establish a persistent connection to a forwarder that the clients can reach directly.*

layer proxy, which we describe in Section 3.4.4.

3.4.3 Advertising Perspectives

Suppose that a Blossom forwarder wants to provide access to resources to which it has access but the rest of the Internet does not. For example, a NAT device may stand between the forwarder and the global Internet. The first task of the forwarder is to establish a *persistent connection* to a forwarder on the outside, so that bidirectional communication across a tunnel will be possible (see Figure 3.4).

Next, the forwarder must advertise its existence to the Blossom directory server(s) in the remote transport domain. The forwarder can accomplish this in one of two ways:

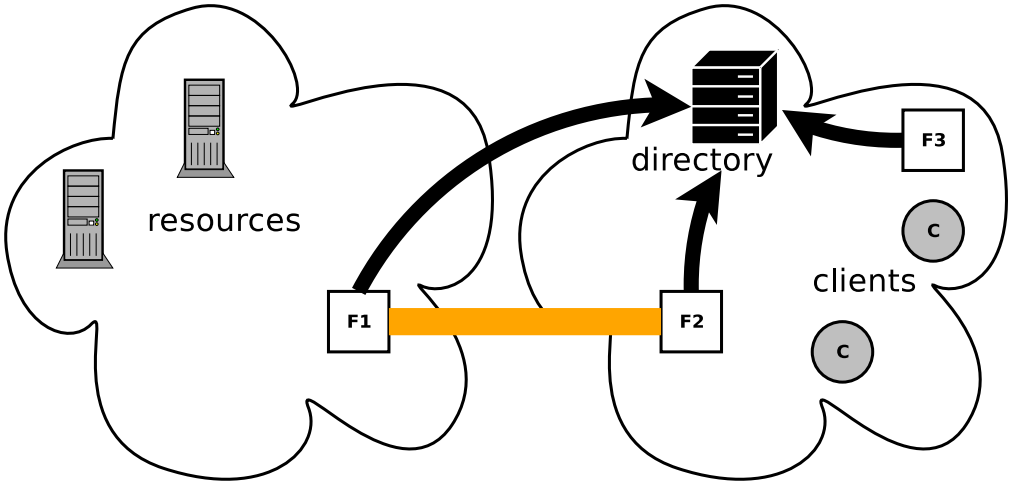


Figure 3.5: Publish to Remote Directory Server. *Once a persistent connection has been established, F_1 may publish to a directory server in the vicinity of F_2 indicating that clients in the vicinity of that directory server should use F_2 to reach F_1.*

- Approach 1. Directly push the descriptor to a directory server in the other transport domain. This approach works particularly well if the other transport domain is "the Internet" and if there are hard-coded, well-known directory servers in "the Internet." The forwarder has the responsibility to inform the directory server about which forwarders can be used to reach it, i.e., to which forwarders it has established a persistent connection (see Figure 3.5).

- Approach 2. Push the descriptor to a directory server in the same transport domain. This is the easiest solution for the forwarder, but it requires the existence of a directory server in the same transport domain that is capable of communicating with directory servers in the remote transport domain. For this to work, some individual Tor forwarders (possibly the directory servers themselves) must have published their descriptors in remote transport domains (i.e.,

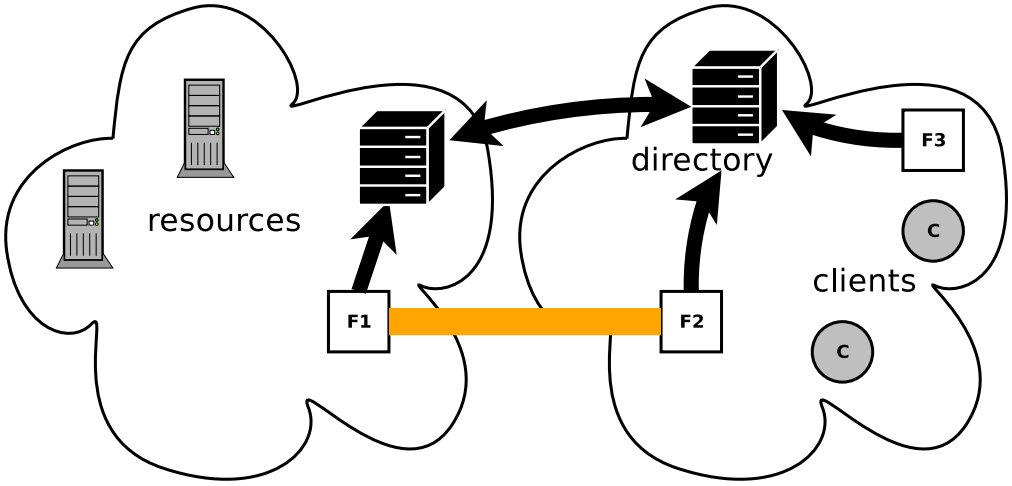


Figure 3.6: Publish to Local Directory Server. *If a directory server exists in the vicinity of F_1, and that directory server exchanges records with a directory server in the vicinity of F_2, then it may be sufficient for F_1 to publish to its local directory server rather than directly publishing to the directory server in the vicinity of the clients.*

followed the first option) to provide a link by which the directory servers can communicate bidirectionally (see Figure 3.6). That is, in Figure 3.6, some forwarder in the left-hand transport domain would need to publish its descriptor to a directory server in the right-hand transport domain so that the directory server in the right-hand transport domain can use that forwarder to contact the directory server in the left-hand transport domain.

Once the directory servers have received reachability and perspective information from the forwarders, the clients can then contact the directory servers to learn how to build paths to the resources (see Figure 3.7).

If all directory servers are within the same transport domain, then Approach 1 is sufficient: forwarders can exist within multiple transport domains, and as long as the

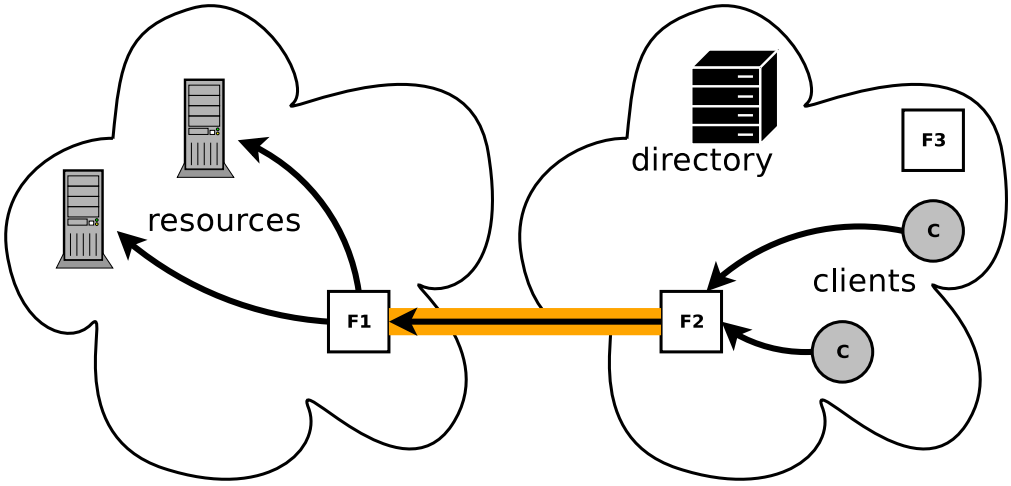


Figure 3.7: Clients Can Now Access Resources. *Once F_1 and F_2 have published their descriptors successfully to the directory service, the clients can use the bidirectional tunnel to access the resources from the perspective of F_1.*

network of transport domains is fully connected by cross-domain links, any forwarder will be able to access any other forwarder in a foreign transport domain simply by extending along the path specified by the directory server. However, we want the system to be truly decentralized, which means not electing any particular transport domain to be the master domain in which entries are published.

3.4.4 Transport Layer Interface

Blossom forwarders are effectively Tor forwarders specially configured to take advantage of the Blossom directory infrastructure. The fact that Blossom clients may enjoy some degree of anonymity by virtue of the fact that Tor provides access to a powerful anonymity tool is certainly beneficial, but it is not strictly necessary to the goals of Blossom. In fact, there are many possible implementations of Blossom, and

the Tor-based implementation seems to be the most reasonable because (a) Tor is a convenient, well-designed, fully-implemented overlay network that is adaptable to many goals, (b) Tor meets more of the requirements outlined in the previous section than any other system, and (c) Tor has real-world users and traffic, which means fewer bugs and the ability to perform live tests.

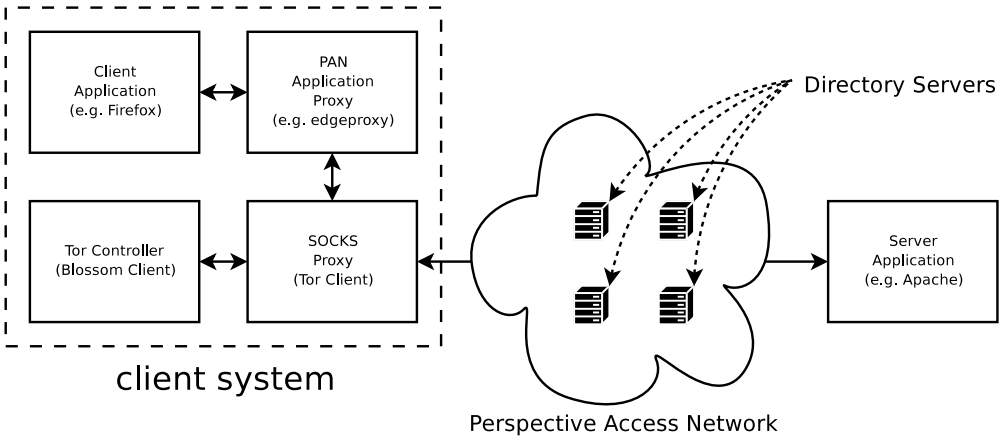


Figure 3.8: Client Perspective Diagram: Blossom. *How Blossom components are integrated with Tor components.*

Clients

Blossom allows users and applications to specify perspectives by appending a metadata query (see Section 3.3.2) to hostnames. Consider the following example (refer to Figure 3.8). Suppose that a user wants to access a web server from a perspective in Greece. The user interacts with a web browser configured to use an application-layer proxy (e.g., Privoxy, or Privoxy enhanced to rewrite application-layer headers), which in turn passes the traffic to the Tor client via the SOCKS protocol. The Tor client passes the SOCKS request to the Blossom client, which

parses the SOCKS request to separate the hostname from the perspective request. The Blossom client then uses Tor and the directory servers within the Perspective Access Network to construct a circuit to an exit forwarder in Greece. Once the circuit has been built, the Blossom client instructs the Tor client to attach the application stream to the newly constructed circuit. Then, the client application may carry out a complete TCP session with the web server via the circuit (from the perspective of the web server, the exit forwarder in Greece is the client).

One of the Blossom diagnostic tools is a dynamically-generated web page cataloguing the various Tor forwarders by country[8] and exit policy[9]. We have also constructed a web interface that allows a user to specify a URL and a country from which she wants to view the URL.[10]

Blossom is a process that manipulates Tor, and it is implemented as a Tor Controller, according to the specification provided in the Tor documentation included with the source package.

Forwarders

A Blossom forwarder needs several capabilities that Tor forwarders generally lack:

- the ability to open persistent connections,
- the ability to know whether to use a persistent connection to reach another forwarder,

[8]Geolocation data are derived from the WHOIS database and are not perfectly reliable.

[9]Tor Network Status, `http://serifos.eecs.harvard.edu/cgi-bin/exit.pl`

[10]Blossom Web Interface, `http://serifos.eecs.harvard.edu/cgi-bin/blossom.pl`

- the ability to define a set of forwarders to which to establish persistent connections,
- the ability to tell a directory server that it has Blossom functionality,
- the ability to tell a directory server that it can be reached via some specific path, and
- the ability to define and enforce policy to manage the set of circuits in which it participates as a forwarder.

Application-Specific Proxies

Some applications require application-specific proxies to take advantage of Blossom. Applications in this category include any applications that send network-layer identifiers inside the transport-layer payload. Web browsing is one such application, so we implemented an HTTP proxy to insert between a web browser and Tor that allows us to use Blossom to browse HTTP.[11]

Our proxy has three primary functions:

First, the proxy parses HTTP requests from the client. The HTTP Host field (45) contains the hostname specified by the client. Since Blossom uses the hostname to express queries, this field contains a string that incorporates the query as well as the name of the host to be requested by the exit forwarder. So, the proxy removes the query component of the string, leaving only the hostname. This is important functionality, since not doing this would confuse many HTTP servers that use vir-

[11]HTTP proxy that rewrites headers and HTML, `http://afs.eecs.harvard.edu/user/goodell/etc/edgeproxy`

tual hosts. The proxy handles both `GET` and `POST` requests, which are functionally different.

Second, the proxy parses HTTP headers in the response and rewrites any redirections to include the metadata query provided in the original request.

Third, the proxy parses HTTP responses received from the server (via Tor and Privoxy in our setup). If the MIME type of the response is anything but `text/html`, it just returns them to the web client. If the MIME type is `text/html`, then it parses the HTML (145) and appends the appropriate metadata query (if one was specified in the request) to each `A`, `FORM`, `FRAME`, `IMG`, and `LINK` tag that contains a hostname that does not include a metadata query suffix. This functionality is useful because (a) images and some redirections refer to content that should be viewed from the same perspective, and (b) most humans browsing web pages like to be able to click on links that should be viewed from the same perspective. Unless the links incorporate the metadata query, there is no guarantee that Blossom will attach those requests to circuits with the correct perspective. In particular, the circuit may break between successive requests, or there may be multiple circuits available simultaneously, possibly offering different perspectives.

The proxy is essentially a proof-of-concept implementation, but it is entirely usable for casual browsing and seems to be generally functional and stable.

3.5 Authentication

Next, we describe a general way of supplementing Perspective Access Networks with a client-to-perspective authentication scheme that can be used as a basis for

access control. This extension to PAN allows a host offering a perspective to require that clients authenticate themselves, just as VPN hosts might similarly require clients to authenticate.

Indeed, in the core of the Internet today, names used to identify resources are universal: they depend only upon the resource and are not defined by the name, physical location, or logical location of the entity requesting the resource. An expectation of universal naming is inappropriate for the Internet; it is both inevitable and beneficial for names of local significance to emerge. We believe that by relaxing the universal naming constraint we can achieve a considerably more flexible network.

However, another way to view Blossom is to consider that it provides a means of describing resources not generally considered to be part of the Internet because they cannot be named, e.g., network services running behind a NAT or firewall. Blossom may introduce some risks in an environment that exhibits a dependence upon a lack of universal naming (or network-layer access) for security.

The benefit of Blossom is its separation of access policy from network-layer mechanisms. Consider an organization whose core IT staff makes network policy decisions regarding external access to internal resources or internal access to external resources. Without Blossom, specific managers and groups have three choices:

- Convince the core to make specific provisions for access policy changes affecting services in their area,
- Convince the core to work with them to deploy special infrastructure allowing partial delegation of the management of network access privileges, creating added complexity, or

- Break network access mechanisms (e.g., punch holes in firewalls, use additional ISPs to provide network uplinks to the core network), potentially undermining the goals of the core administrators.

Blossom provides organizations the opportunity to delegate responsibility for network access policy to a broader set of managers capable of making policy decisions. By providing a mechanism that can be managed locally but verified centrally, we alleviate some technical barriers (e.g., firewall configuration mechanisms) to defining policy. Ultimately, technology should be used to facilitate management decisions, not encumber them. Individual managers can make executive decisions about whether allowing access to a particular resource is consistent with the stated objectives of the organization or not. We seek not to answer the question of whether such empowerment is appropriate in each individual case, but only to ensure that the requirements of particular network technologies do not prevent such questions from being asked.

Many enterprises use end-to-end authentication for some services, but there are a number of popular services that rely upon the assumption that the only hosts that have access to the service are physically on the same LAN or have particular network-layer addresses. For example, the market for secure fileservers is small. We suspect that this means that most distributed filesystems used by most businesses base their security upon assumptions about how clients are connected. We do not seek to create new risks for organizations that rely upon firewalls; we seek to provide a means by which firewalls need not unnecessarily constrain access to services. This is a problem that bridges the gap between IT and management, and our solution must respect the interests of both sides.

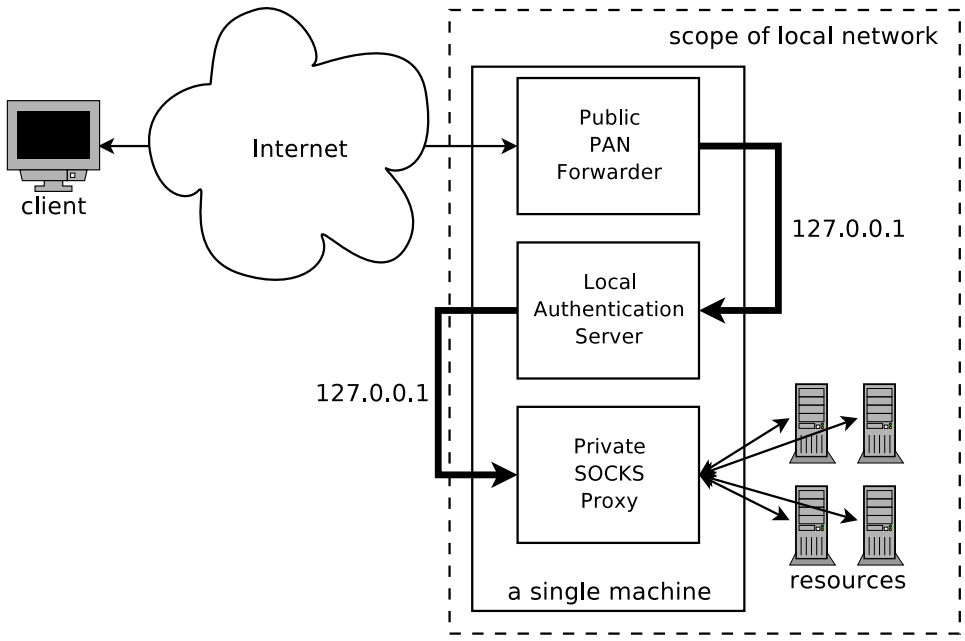


Figure 3.9: PAN Client Authentication. *To authenticate clients, stipulate a restrictive exit policy that exclusively allows access to a local authentication service running on the same host as a PAN forwarder. Upon successful authentication, open a tunnel through the authentication service to a private SOCKS proxy. The client can now use this SOCKS proxy directly to access resources local to the PAN forwarder.*

Ultimately, we seek to provide a means by which businesses can extend the trust envelope to include not only hosts who happen to be situated on the local network but also authenticated parties from the outside as well, in a similar manner to a VPN but with the semantic features of a perspective. The openness of a PAN does not conflict with good security practice, and we propose a simple strategy for integrating PAN into environments in which some services rely upon the presumption that local users are legitimate:

- For secure services, we simply configure our Blossom forwarder to exit to the corresponding IP address(es) and port(s).

- For services that are insecure because they potentially send or receive sensitive data in the clear, providers of the service may derive benefit from running the Blossom forwarder on the same machine with the service. Using onion routing, we get an end-to-end encrypted tunnel from the client to the machine with the service at no additional cost.

- For services that are insecure because they do not provide authentication, we provide authentication on the side with the Blossom exit forwarder via the following technique. The Blossom forwarder provides access to only a single TCP port on an authentication server collocated on the Blossom forwarder itself. Listening on this port is a service that provides cryptographic authentication (e.g., via SSH or SSL), and we use the resulting secure channel to open a secure tunnel from the client to a SOCKS proxy running on the authentication server. The client can then use SOCKS to communicate with arbitrary TCP services in the network containing the Blossom forwarder (refer to Figure 3.9). This style of authentication is useful only to exit forwarders, since only exit forwarders provide access to application servers.

3.6 Practical Applications

In this section, we catalogue some potential applications for Perspective Access Networks. We argue that present-day scenarios are sufficient to make use of the various features of PAN, including both its path-vector routing infrastructure and its support for circuits that contain multiple forwarders.

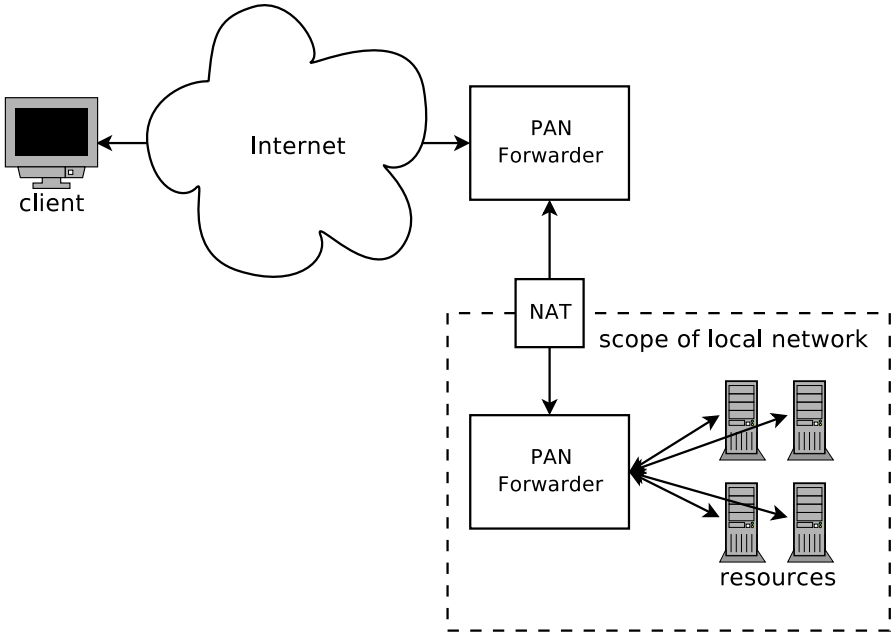


Figure 3.10: Direct Access to Server Restricted. *PAN forwarders located behind NAT devices must "reach out" to establish persistent connections with other PAN forwarders.*

First, consider Figure 3.15, which illustrates a case in which an enterprise uses the Internet to connect its various offices. Note that resources within one network can implicitly use PAN to refer to resources accessible via a PAN forwarder in a remote office, even if such resources are not available directly.

In each of these examples, the functionality that PANs provide is similar to the functionality that VPNs provide, except that PANs also provide a *directory service* and a *routing infrastructure*. Section 1.3.3 elaborates upon the distinction.

3.6.1 Circuits with Multiple Forwarders

Figure 3.10 illustrates a scenario in which the PAN exit forwarder resides behind some sort of NAT (or firewall). The NAT imposes a unidirectional link in which

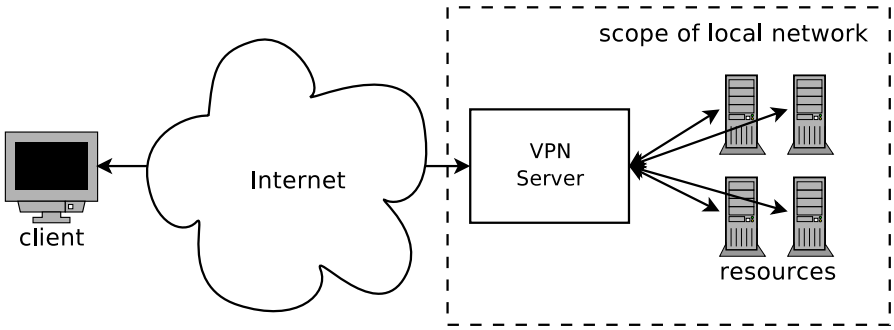


Figure 3.11: VPN Server. *Most deployed VPN servers are directly accessible by clients in most locations.*

hosts on the "outside" of the NAT cannot initiate communication with hosts on the "inside" of the NAT. Since clients cannot access the exit forwarder directly, the forwarder itself must first establish a connection to some other PAN forwarder on the outside, creating a bidirectional link through which new TCP connections can be carried in either direction. In effect, the PAN forwarder outside the NAT becomes a rendezvous point that clients can reach. The connection **must** be persistent and **must** be re-established from the inside out in the event of NAT failure, or else the bidirectional link will be severed and it will no longer be possible for new connections to the internal forwarder to originate from outside the NAT. Note that this scenario requires a two hop path: the first hop is the rendezvous point, and the second hop is the target forwarder. Note that this is different from the case of the directly accessible VPN, in which the VPN server itself does not need to reach out to be directly accessible by clients (refer to Figure 3.11).

Figure 3.12 illustrates a scenario in which an active adversary prevents a client from accessing a certain subset of the PAN forwarders. This subset may possibly contain PAN forwarders that the client wants to use as exit forwarder in its circuits.

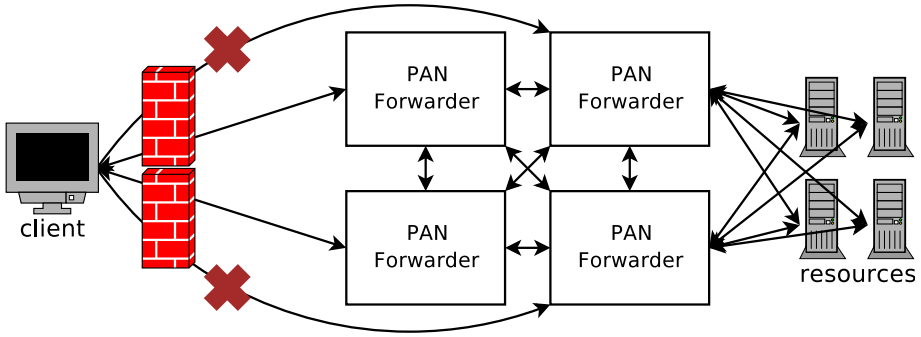


Figure 3.12: ADVERSARIAL FILTERING. *Adversaries may eliminate the ability for clients in particular regions of the network to connect to certain forwarders.*

If a client cannot access a particular forwarder directly, then the client must build a circuit consisting of multiple forwarders. If we assume that there is only one adversary, and that this adversary acts in the vicinity of the client, then a two-hop path will be sufficient. However, if we assume a more powerful adversary, or multiple adversaries, that block communication between arbitrary pairs of forwarders, then a path of length greater than two may be necessary.

Note that Figure 3.12 implicitly presumes one of two possible adversary models, both of which necessitate a multi-hop path. In the first model, the adversary wants to block PAN forwarders but is insufficiently powerful to do it effectively, and may only block some subset of the forwarders at one time, perhaps because of the costs of updating routers to block PAN forwarders, perhaps because of the costs involved in maintaining an up-to-date list of PAN forwarders, or perhaps because of the costs involved in blocking PAN forwarders that offer other important services are too much to bear. In the second model, the adversary is quite powerful but does not want to block PAN forwarders in particular. Specifically, the adversary blocks networks (for whatever reason, including blacklisting, objectionable content reports, accident, etc.)

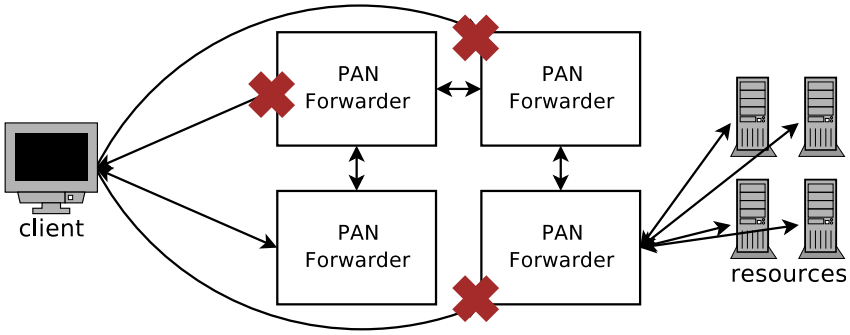


Figure 3.13: Routing by Policy. *Locally-configured policies maintained by individual PAN forwarders may constrain the construction of circuits.*

without intending to block PAN, but the PAN forwarders are blocked as "collateral damage."

3.6.2 Routing

Neither of the examples described in Section 3.6.1 require a particularly sophisticated routing infrastructure; in both cases, we could assume a central directory server that serves as an index, mapping particular forwarder names or metadata entries to ordered lists of forwarders that could be used as circuits that provide access to the exit forwarders in question. However, there are a number of reasons why a general-purpose routing framework is a critical feature of Perspective Access Networks:

- Scalability and Robustness. PAN provides a flexible infrastructure that allows clients to learn how to construct circuits of arbitrary length, containing arbitrary forwarders. In this manner, paths from a client to a particular forwarder may be dynamically changed in the event that some individual links are broken. Also, a general routing infrastructure alleviates dependence upon a

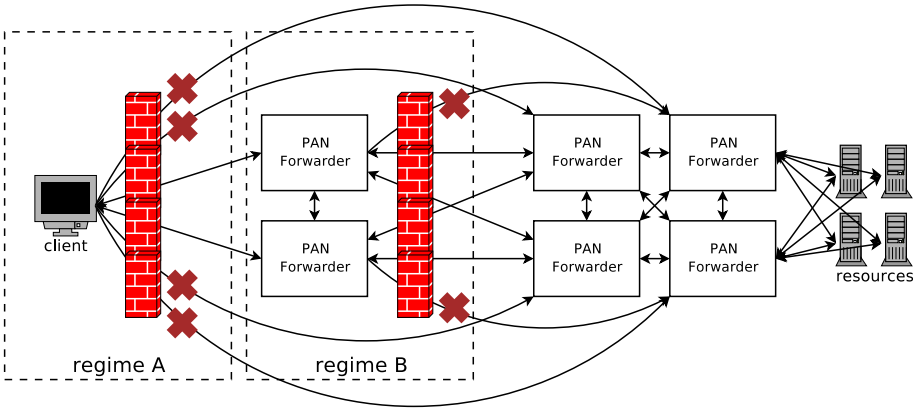


Figure 3.14: Multiple Adversaries. *Network constraints imposed by adversaries in different regions of the network may necessitate the creation of longer circuits.*

single authority for ordaining all paths used by clients throughout the network.

- Routing by Policy. Figure 3.13 illustrates a scenario in which policies configured on individual PAN forwarders force the client to build a multi-hop circuit through a set of PAN forwarders. Individual forwarders may be configured such that they do not accept connections from particular clients. Also, individual forwarders may be configured such that they can only extend circuits to other forwarders from a specific set or such that particular pairings of prior and subsequent forwarders in a circuit are disallowed.

- Multiple Adversaries. Figure 3.12 presents a situation that makes use of a two-hop path in the presence of a single filter that restricts access to some part of the network. However, large-scale networks may contain many potential adversaries with complex interests, and various parties controlling different parts of the network infrastructure may have deployed filters in such a manner that the only legitimate circuit from a client to a particular forwarder may require

more than two forwarders. Figure 3.14 illustrates a network with several control points at which different filtering mechanisms are enforced: regime A only allows outbound traffic to regime B, and regime B does not allow outbound traffic to the particular forwarders that a client wishes to use. The resulting situation requires the client to use a three-hop path to reach its chosen forwarders.

- A "Coreless" Internet. All of our discussion in this section has been predicated upon the idea that there is some part of the Internet that is central, in the sense that everyone can agree about its role as "core," and in fact with strikingly few exceptions, the Internet today does have a fairly well-defined core. Despite regimes in Asia that filter access to democratic speech, regimes in Europe that filter access to hateful paraphernalia, and regimes in America that filter access to intellectual property, Internet users do have an expectation that for the most part, the filters are close to the edges, and the core of the Internet is mostly uniform. Nonetheless, one of the key design goals of PAN is to allow access to resources even if the filters are in the center of the network. There are numerous arguments that suggest that the breakdown of a common "core" may arise in the future, and in this event, a general-purpose routing infrastructure will be necessary to assure access across different parts of a fragmented network.

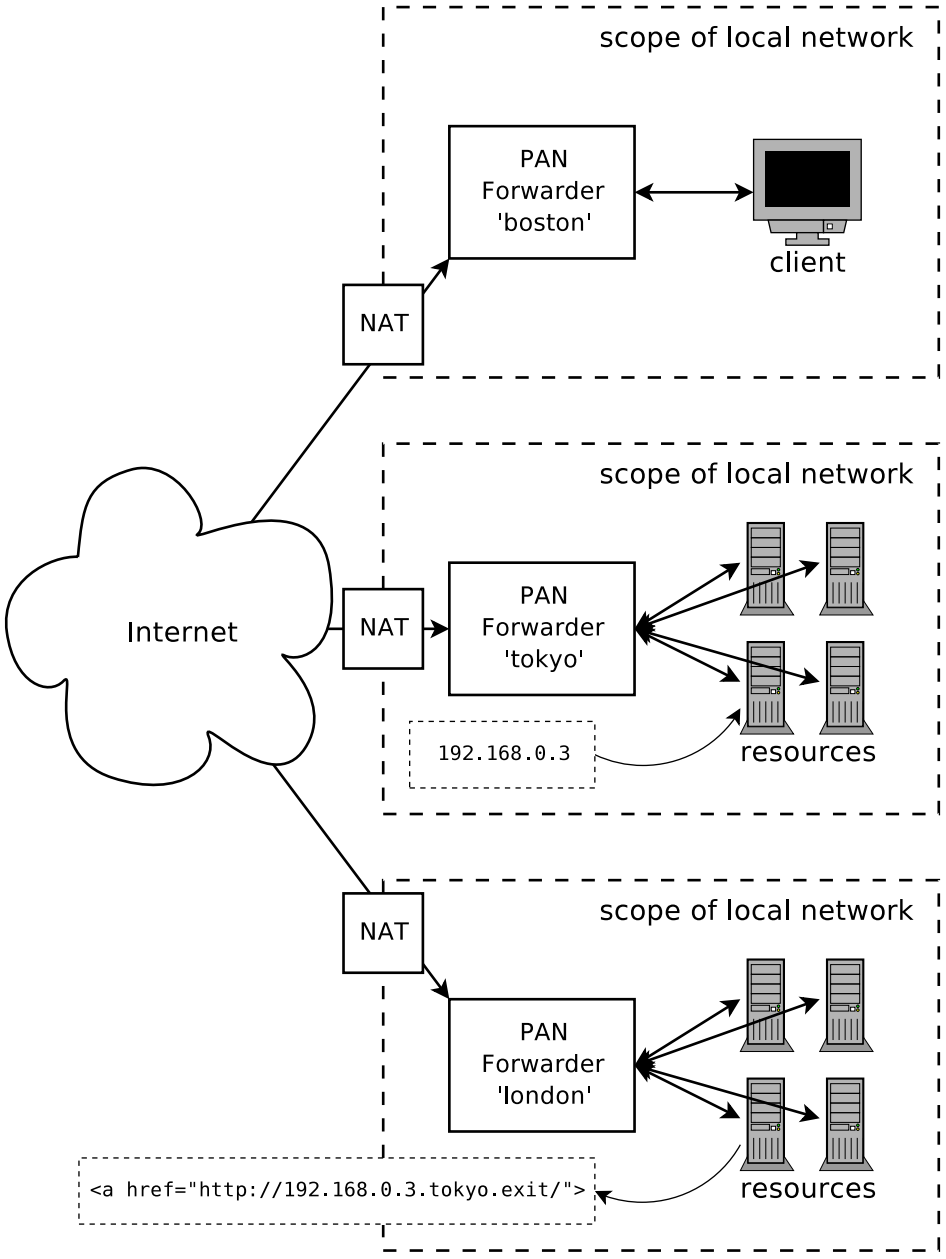


Figure 3.15: Explicit References across Boundaries. *Resources located in one office of an enterprise can refer to resources only accessible from within other, specific offices.*

Chapter 4

Directory Service

This chapter focuses on the directory service that enables clients to access perspectives within Perspective Access Networks. Clients consult directory servers to learn the paths that lead to perspectives of their choosing.

For the PAN overlay, we assume only that each forwarder has the ability to communicate bidirectionally with some subset of the other forwarders. A PAN client that wishes to view a resource from the perspective of a particular forwarder F uses the PAN distributed directory service (provided, in our prototype, by a subset of the forwarders) to determine a path of connectivity through the forwarders to F. The PAN client then constructs a source-routed circuit through the forwarders on the path to F, which then performs a DNS lookup to resolve the local resource name to an IP address from its point of view and accesses the resource on behalf of the client. The client therefore accesses the resource from the perspective of F. (Figure 4.1 provides a conceptual overview.)

Since we do not require a global unique naming scheme for resources, we need

a way to uniquely identify a resource. We therefore require forwarders to generate unique, self-certifying identifiers and a PAN client specifies a particular resource by concatenating the forwarder ID with the resource name as resolved by the forwarder. This design choice, however, sacrifices a certain amount of aggregation we can perform when advertising forwarder route information within the PAN overlay.

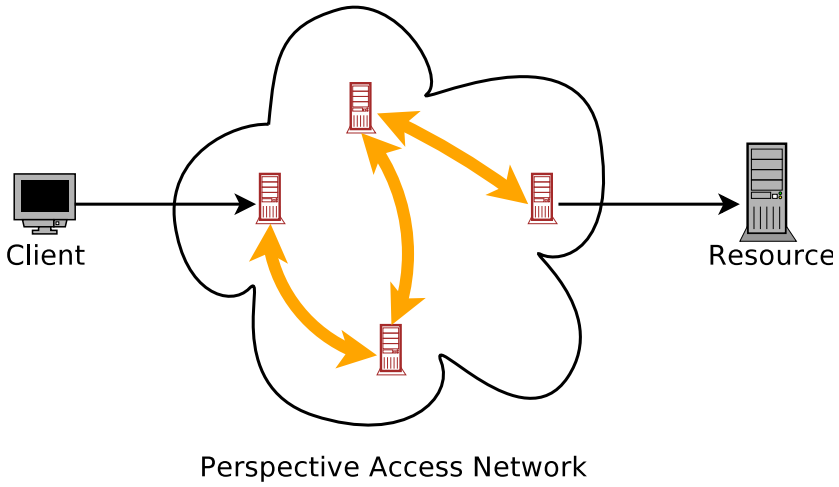


Figure 4.1: Perspective Access Network Overview. *PAN presents a peer-to-peer network for sharing perspectives, allowing access to resources in circumstances in which the meaning of names and addresses is a function of their context.*

This chapter is arranged into four sections, as follows. Section 4.1 presents the directory service architecture in detail. Section 4.2 illustrates the most significant design tradeoffs, including decisions surrounding client queries and perspective propagation. Section 4.3 describes the general configuration of individual directory servers, including peering arrangements. Section 4.4 closes the chapter with a detailed discussion of the policy framework for managing which perspectives to accept and propagate; our discussion includes some useful examples.

4.1 Directory Architecture

PAN consists of a pairwise-connected overlay network of *forwarders*, each of which has access to some set of Internet resources. Some resources may be available to some nodes but not others. The overlay network that connects all of the forwarders to each other includes a *data plane* that carries tunnelled DNS requests and TCP sessions, as well as a *control plane* that carries routing information.

There are a number of problems with a distributed approach to assigning names in a network. For example, two network components may have the same name, and there are performance costs associated with choosing names that do not inherently carry location information. However, we suggest that it is both possible and beneficial to sacrifice universal naming by allowing access to resources whose names are locally governed.

To address the concern about uniqueness of names used to identify forwarders, we allow each forwarder to generate a self-certifying identity (such identities may be mapped to human-readable nicknames by third-party certification authorities). Each forwarder, then, possesses two names: a *global* name, used to identify itself within the PAN network, and a *local* name, used to identify itself within its local namespace. By considering that each forwarder provides access to resources within its own local namespace, we avoid requiring that all names for all Internet resources be globally unique.

To specifically identify each Internet resource, we combine the locally meaningful name of the resource (e.g., a hostname such as `www.google.com`) with an identifier specifying the name of the forwarder from which we want to access that resource (e.g.,

the self-certifying name of a forwarder, like `89dc1c13`). For the purpose of Blossom, we assume that resources are named by hostname or IP address, so to access a resource listening on TCP port `80` of `192.168.0.3` as seen by a forwarder named `89dc1c13`, we would represent the resource as `192.168.0.3.89dc1c13.exit:80`.

Some PAN forwarders also serve as directory servers, and every PAN directory server is also a forwarder. Each directory server provides a set of *records*: (a) a *master record*, containing attributes describing itself, (b) a set of *directory records*, each containing attributes describing directory peers, and (c) a set of *forwarder records*, each containing attributes describing individual PAN forwarders. The directory server publishes these records by responding to queries in the form of HTTP-GET requests, and these attributes are periodically pushed to neighboring directories via directory updates in the form of HTTP-POST requests. Figure 4.2 illustrates one possible set of records stored in a directory server given one possible network of directory servers and standalone forwarders.

4.1.1 Master Records

A complete PAN directory server listing includes exactly one *master record*, which contains three attributes, as follows: a *header* consisting of the name of the directory server and its version, a *timestamp* indicating when this directory listing was created, and a *status* record identifying each forwarder indexed by the directory, including a bit that indicates whether the directory believes that forwarder to be active. The bit specifying whether a given forwarder is reachable is set to true when the directory server receives a sufficiently recent descriptor for an individual forwarder, and it is

set to false when the descriptor expires.

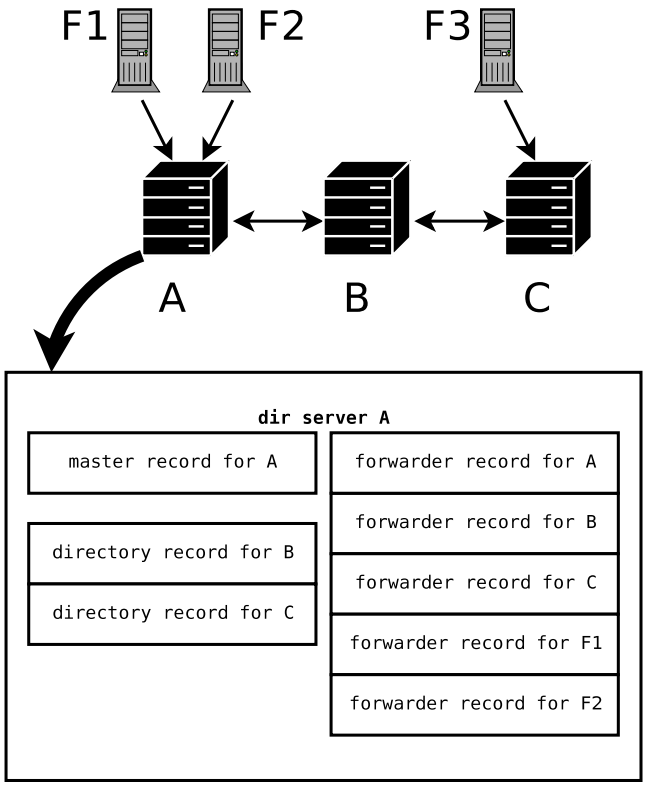


Figure 4.2: Records in PAN Directories. *Given three directory servers A, B, C and two standalone forwarders F_1, F_2 as shown above, the table illustrates one possible set of records published by directory server A.*

4.1.2 Directory Records

Each PAN directory server publishes a number of *directory records*, each containing a set of attributes that describe a specific peer directory server. A directory server accrues a set of directory records over time via directory updates from its neighbors. Unlike peer-to-peer filesharing services such as Gnutella or BitTorrent, PAN is designed with the goal of balancing scalability with minimization of connection setup

latency for clients connecting to services. Thus, clients do not request forwarder records via broadcasting or heuristic searches; instead, each directory maintains a set of directory records, each uniquely corresponding to one of its peers. Scalability dictates that each individual directory server need not know everything about the entire network, so there is no guarantee that each directory server contains a record for each other directory server in the entire network.

When a client issues a query for a forwarder record, but a directory server has no corresponding forwarder record, the directory server may refer the client to a set of directory servers that have previously indicated knowledge of forwarder records matching the request of the client. This *referral* consists of a set of directory records and the forwarder records that correspond to the directory servers.

Since directories are not required to explicitly fetch information on behalf of their clients, a client that queries a directory for information can expect to be referred to a specific neighboring directory server. However, such referrals are not arbitrary: clients seeking a particular forwarder record will be sequentially referred to some subset of the set of directories along the reversal of the path by which the advertisement of the forwarder propagated through the network.

We use ABNF (33) to specify the format of text fields. We specify self-certifying forwarder names and metadata fields according to the following formats:

```
FNAME := 40(ALPHA / DIGIT)                    (4.1)

FMETA := *(ALPHA / DIGIT / "-")               (4.2)
```

Each directory record contains the following attributes (refer to Table 4.1 for the

field name	*format*
Service Designation	`*VCHAR`
Propagation Path	`*1FNAME *("," FNAME)`
Summary	`FNAME "=" *DIGIT *("," FNAME "=" *DIGIT)`
Compiled Metadata	`FMETA *("," *FMETA)`

Table 4.1: Directory Record Field Formats.

ABNF representation of the field formats):

- Service-Designation. *(required)* This field tells a client how to connect to a directory server, given that the client has already constructed a circuit to the forwarder residing on the same machine as the directory server. In our present implementation, this field is a TCP port number.

- Propagation-Path. *(required)* This field contains an ordered list of directory servers (starting with the origin) through which this particular directory record has propagated before reaching the directory server upon which it presently resides. The primary purpose of this field is to avoid cycles in the propagation of directory records.

- Summary. *(optional)* This field provides a list of PAN forwarders associated with this particular directory record, indicating that the corresponding directory offers to forward traffic to the indicated set of PAN forwarders. For each forwarder in the list, this attribute also includes *one possible forwarding path* leading to that forwarder. Note that descriptors for the forwarders indicated in this list may or may not be published at the particular directory server. See section 4.1.4 for details.

- COMPILED-METADATA. *(optional)* Propagation of metadata is analogous to propagation of individual forwarder descriptors. This field is a list of metadata strings (i.e., perspective attributes) representing the union of all of the *Metadata* attributes corresponding to all of the forwarders that appear in the *Summary* field of this directory record. For each *Metadata* attribute, this attribute also includes *one possible forwarding path* leading to a forwarder whose perspective has that attribute. Therefore, directory servers **may** issue referrals to clients querying for forwarder records matching some particular metadata field in the same manner by which they **may** issue referrals to clients querying for specific forwarders by name.

As an optimization, a PAN client **may** use the forwarder-specific or perspective-specific forwarding path information in *Summary* or *Compiled-Metadata* fields, respectively, to build circuits toward a given forwarder or perspective without querying directory servers along the path (provided that the client has access to sufficiently recent descriptors for the constituent forwarders). This can potentially improve circuit setup latency, but there are tradeoffs as well. First, a client choosing this option does not receive information about possible alternative paths, thus waiving its option to choose its path from the set of advertised possibilities. Second, the path is not actually guaranteed to work; inconsistency resulting from slow routing convergence may allow forwarding paths that are no longer applicable to persist for some time in the *Summary* and *Compiled-Metadata* fields offered to clients by directory servers.

field name	*format*
Forwarder Descriptor	(determined by substrate descriptor format)
Propagation Path	`*1(FNAME *("," FNAME))`
Forwarding Path	`*1(FNAME *("," FNAME))`
Metadata	`FMETA *("," *FMETA)`

Table 4.2: Forwarder Record Field Formats.

4.1.3 Forwarder Records

When a PAN forwarder publishes its descriptor, metadata, and connection information to some directory server, the directory server in turn creates a forwarder record using that information. Each forwarder listed in a directory has exactly one corresponding forwarder record. In general, forwarder records are updated more frequently and propagated less widely than directory records; see Section 4.1.5 for details. A directory server **must** publish a forwarder record for itself. Each forwarder record contains some subset of the following fields (refer to Table 4.2 for the ABNF representation of the field formats):

- Forwarder Descriptor. *(required)* PAN directory servers provide *descriptors* that can be used by the PAN client to establish circuits through the forwarding network. Descriptors are self-signed statements published by forwarders that contain contact information, including IP address and port for accepting circuit-building connections, public key, and salient information about the capabilities of the forwarder, including exit policy and bandwidth measurements.

- Propagation-Path. *(required)* This field contains an ordered list of directory servers (starting with the origin) through which this particular forwarder record

has propagated before reaching the directory server upon which it presently resides. The primary purpose of this field is to avoid cycles in the propagation of forwarder records. The value of this attribute may be empty, in which case the propagation path for this particular forwarder record is presumed to be the empty list (i.e., the forwarder described by this record published its information directly to the directory server upon which this record presently resides). Note that this path is not necessarily the same as that provided by the *Forwarding-Path* attribute (described next).

- FORWARDING-PATH. *(required)* This field contains an ordered list of forwarders indicating the circuit that a client should construct to reach the forwarder described by this record, starting with the forwarder closest to the current directory server. Differences between this list and the list provided by *Propagation-Path* attribute arise in two ways. First, directory servers through which a forwarder record propagates are not required to add their names to the forwarding path. Second, the PAN architecture allows forwarders to publish their descriptors in directories in locations from which those forwarders are not directly accessible; to address this, the forwarder may provide instructions by which clients can reach it from the perspective of the directory to which it publishes its information. These instructions appear in the form of a *forwarding path*, a particular sequence of forwarders to which to connect to establish a circuit including the target forwarder; see Section 4.1.5 for details.

- METADATA. *(optional)* This attribute provides additional perspective information (e.g., geographic region, network name, connectivity information, access to

particular resources, etc.) describing the forwarder.

4.1.4 Client Interaction

Our implementation of PAN leverages the circuit-building module of Tor (38) to instruct a running Tor process to build a circuit through the overlay of PAN forwarders. (Tor provides its own directory service, but PAN does not make use of it.) To see how the various components interact, refer to Figure 4.3. The main PAN client process itself does not interact with client applications directly; instead, it communicates with PAN directory servers using specially-built Tor circuits, and it uses descriptors obtained from these conversations to instruct Tor to build circuits that client applications can use. To take advantage of PAN, client applications may need to interact with an application-specific proxy that assures that requests for network resources are semantically correct. For example, a proxy for a web browser might rewrite HTTP headers to excise the PAN forwarder request from the hostname fields. Similarly, the same proxy might rewrite HTML tags containing URLs to ensure that all links on a page are accessed via the same PAN directives when clicked or loaded automatically.

Issuing Queries

To establish a path to a specified exit point, PAN must first determine the path to the exit point and obtain descriptors for each of the forwarders along that path, including the last one. Sufficient information necessary to learn a path to a given destination and all of the requisite descriptors may be available from the directory

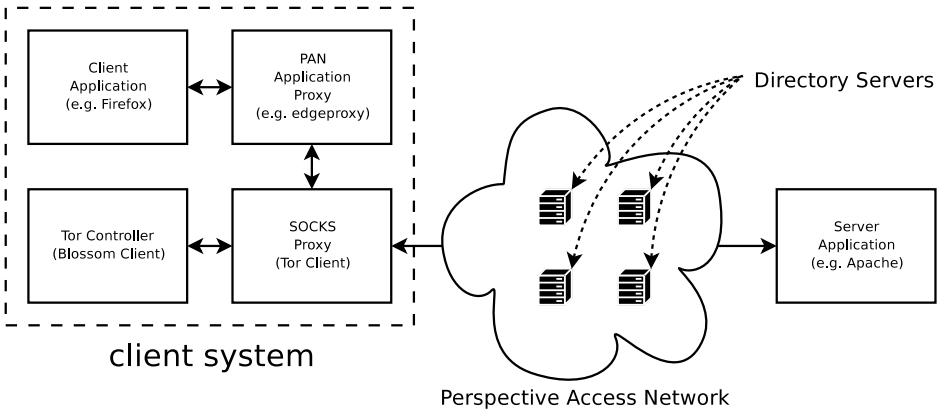


Figure 4.3: Client Perspective. *Client applications communicate with PAN via a series of proxies; PAN consists of software (a program that controls a running Tor process) as well as a service (the perspective access network itself).*

server to which the client speaks directly. Otherwise, the client will need to obtain the missing information via a series of queries to directory servers. See Figure 4.4. Each time that a client queries a directory server A and is referred to another directory server B for more information, the client extends the circuit used to communicate with A to B, thus adding a single hop to the circuit.

There are two types of queries, *explicit queries* and *perspective queries*. *Explicit queries* request a path to a particular forwarder whose name matches a given string, indicating that the client wants to build a circuit that terminates at some specific last-hop forwarder. *Perspective queries* request a path to a forwarder with certain attributes in its corresponding *Metadata* field, indicating that the client wants to build a circuit that terminates at any last-hop forwarder whose forwarder record on some directory server matches some set of criteria. Note that directories control the content of *Metadata* fields within forwarder records, so, for example, a client issuing a perspective query **may** choose to reject a circuit to a specific forwarder if its descriptor

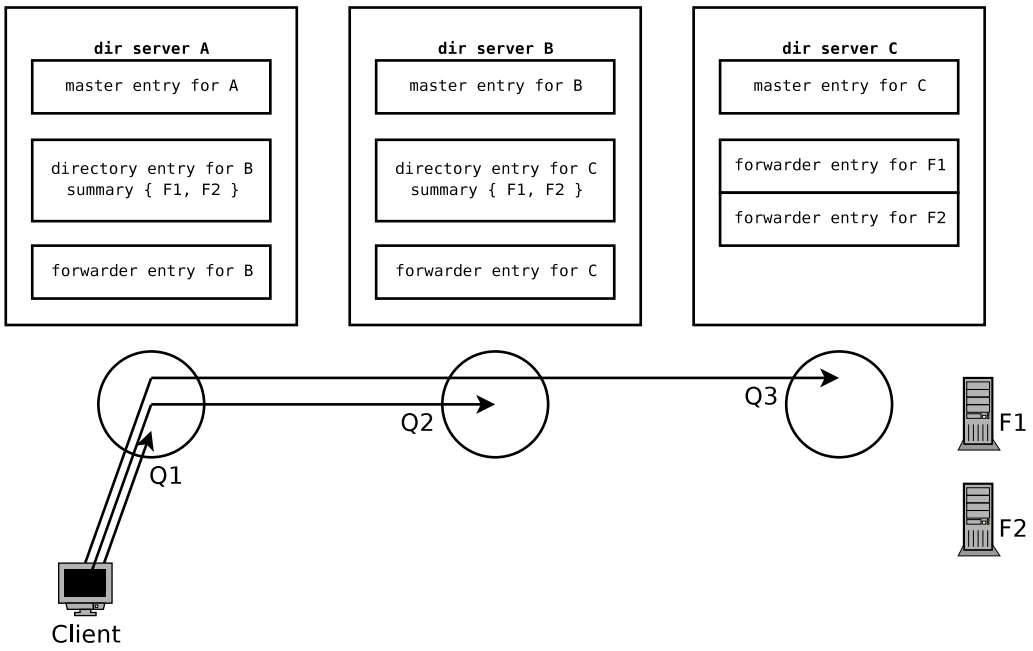


Figure 4.4: Issuing Queries. *Suppose that a client application requests a service as seen by forwarder F_2, and the PAN client is configured to use directory server A. The client first sends a query to A, who responds with a referral to B. The client next sends a query to B, who in turn refers it to C. Finally the client sends a query to C, who has the descriptor. The client then uses the resulting circuit through $\{A, B, C\}$ to extend the circuit to F_2 and connect to the target service via F_2.*

does not contain a metadata record matching the original request.

The contract between a directory server and a client issuing a query is as follows. If a client issues a query, then a response from the directory server **must** include the following:

- (a) a forwarder record for a forwarder that matches the query,
- (b) (in the event of an explicit query) some set of directory records and their corresponding forwarder records, such that each directory record contains either a *Summary* field containing an element that matches a given forwarder name,

- (c) (in the event of a perspective query) some set of directory records and their corresponding forwarder records, such that each directory record contains a *Compiled-Metadata* field containing an element that matches a given string, or
- (d) an empty list of records, indicating that the query was unsuccessful.

Finally, a directory server **may** interpret a query as *recursive*, meaning just as some DNS servers are configured to issue DNS requests on behalf of their clients, PAN directory servers may issue queries on behalf of their clients, provided that they return results that satisfy the criteria listed above. One incentive to configure directory servers to perform recursive queries is that it reduces the amount of work and network activity on the part of the client. However, this comes at the expense of increasing the computational burden and network utilization of the directory server. While such a tradeoff may be useful in an enterprise setting, it is less likely to be useful for arbitrary directory servers accepting public queries.

A client may specify to the directory server that it intends for its query to be non-recursive, in which case the directory **should** honor that request (to avoid the chance that a cached entry might be wrong).

Building Circuits

In our prototype, once it has obtained forwarder records for the entire path to the last-hop forwarder, the PAN client will provide the necessary descriptors to Tor and then ask Tor to build a circuit using those descriptors (see Figure 4.5). Once the circuit has been built, PAN will inform Tor that the TCP stream received from the client application should be attached to the newly constructed circuit. We have used

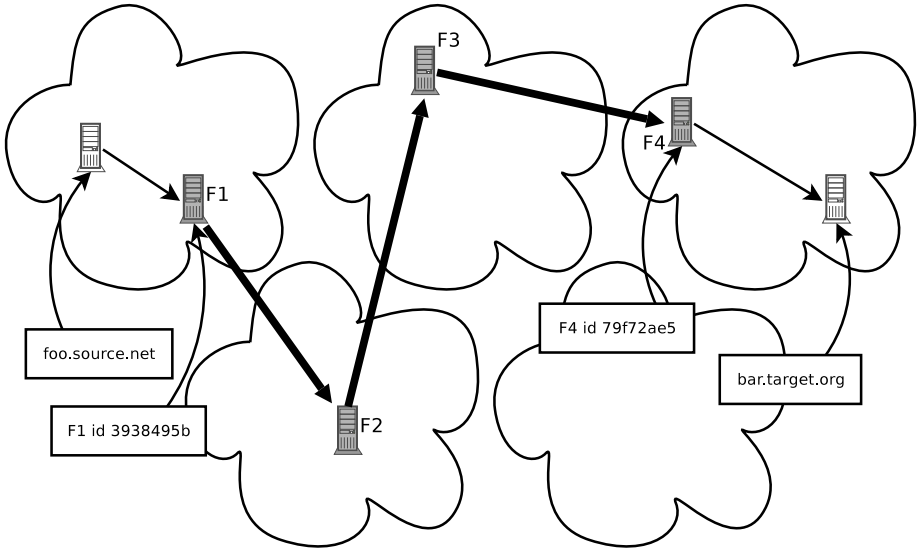


Figure 4.5: Accessing a Resource. *After making use of the PAN directory servers, a client system has a source route suitable for building a circuit through the set of forwarders to the last-hop forwarder, through which the client can access the (otherwise occluded) Internet resource.*

our implementation[1] to confirm that the set of web pages accessible from some ISP in China differs from the set of web pages accessible from some ISP in Boston.

4.1.5 Directory Protocol

The directory servers propagate both forwarder records and directory records to other directory servers throughout the system. In this manner, any client using any of the directory servers throughout the system will have a measure of assurance that it can build a circuit to its requested forwarder, provided that directory server configuration permitted the propagation of routing information.

Directory records are stored as *long-term state* that is assumed to be up-to-date

[1]Blossom, `http://afs.eecs.harvard.edu/~goodell/blossom/`

unless a *Directory Update* request from a neighboring directory server is received. The message volume involved in maintaining synchronicity of routing information can be expensive, so a directory periodically pushes the changes to its neighbors. Reliability is achieved by stipulating that if a directory server A fails to successfully send an update to a particular neighboring directory server B, then A will consider B to be offline. When a directory comes online, and periodically over a long time interval thereafter, it requests a *burst* from each of its neighbors. The burst contains all of the directory records that the neighbor would ordinarily provide via regular directory updates, reflecting the complete state of what the neighbor would ordinarily provide to the directory making the request. After receiving the bursts, the requester applies a path-selection algorithm to determine the set of records that it should propagate, and it updates each of its neighbors with this set of records. Subsequently, the directory will only receive *directory updates* from its neighbors when individual records change. Each time the directory server receives a directory update that results in a change to its own set of records, that directory server **must** notify its neighbors about the change within a reasonable period of time (unless filtering and aggregation rules obviate the need to update a neighbor about the change).

Conversely, forwarder records are stored as *short-term state* that is periodically refreshed, since forwarder descriptors change frequently and individual forwarders themselves may join and leave the network frequently. Individual forwarder records must be periodically re-issued: if a forwarder record becomes too old before it is replaced, then directory servers **should** discard it.

Periodically, neighbors send empty updates to each other, even if they have no

directory changes to send. Such empty updates are *keepalive* messages. If a directory has not heard from one of its neighbors for a sufficiently long period of time, it concludes that the link to the neighbor has been severed and responds by issuing a *withdrawal* message to its peers indicating that the directory record is no longer available. Withdrawal messages carry valid *Propagation-Path* attributes, and any directory server A that currently offers a directory record whose *Propagation-Path* attribute contains the name of a neighbor B from which it received a withdrawal message **must** propagate to its other neighbors either a message announcing the withdrawal of B or an ordinary directory record with a *Propagation-Path* attribute that does not contain B. In this manner, all directory servers that have selected the withdrawn route will be informed of the change (as in BGP, failing to propagate the results of a withdrawal may constitute an attack).

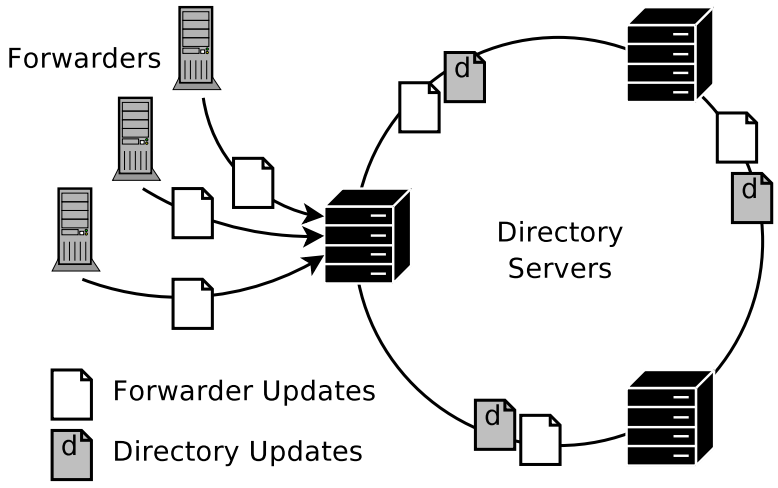


Figure 4.6: Directory Propagation. *Each forwarder publishes its forwarder record to some set of directory servers, and each directory server publishes its directory record to its neighbors. Directory servers propagate both kinds of records according to their individual policies.*

Directory Propagation

Both directory records and forwarder records are propagated using a BGP-like path-vector protocol that includes a simple route selection algorithm applied at each directory server. Figure 4.6 illustrates the process by which route information is propagated through the network. Each forwarder advertises its forwarder record to some set of directory servers, and directory servers propagate the forwarder record through the network as far as policy permits. Forwarders that are also directory servers advertise only to themselves. Each directory server creates a directory record for each of its neighbor directory servers and propagates the record through the network. Thus, forwarders push forwarder records to directory servers, and directory servers push both forwarder records and directory records to other directory servers.

If a directory server receives two conflicting forwarder records (e.g., two records with different attributes for the same forwarder), then it chooses to propagate the one whose *Forwarding-Path* attribute has the shorter length. Figure 4.7 provides an overview of how forwarder information propagates in the general case. The specific configuration of individual directory servers may cause exceptions to these rules; Section 4.3 discusses this in greater depth.

Directory Requests

Directories address five different kinds of requests, all issued using HTTP/1.1 (45) (refer to Table 4.3 for the ABNF representation of the request formats):

- Complete Listing. This is a request for the entire set of records, including its master record, all directory records, and all forwarder records. The response

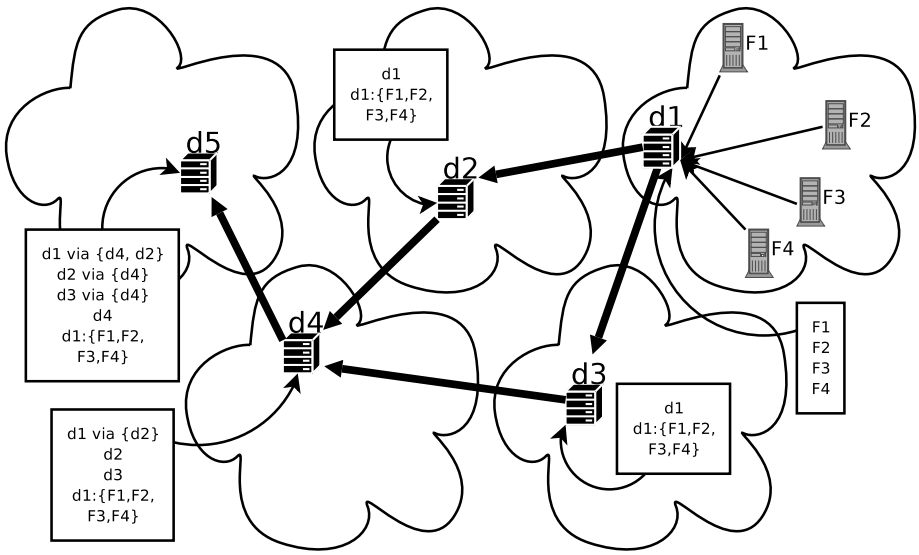


Figure 4.7: Advertising PAN Forwarders. *PAN directory servers use a path-vector algorithm to propagate contact information for PAN forwarders. Black lines indicate the path taken by an advertisement initiated by the directory server labeled d_1. The boxes represent the records stored at the various directory servers, including Propagation-Path and Summary attributes of directory records.*

to this request can potentially be quite large, but query overhead for a client could be reduced substantially if most of the forwarders to which it desires to build circuits have forwarder records published on the same directory server.

- Directory Burst. This is a special request sent by a directory server when it first comes online to bootstrap its knowledge of the records advertised by each of its neighbors. A directory server responds to this request by providing a master record, all of its hard state (i.e., all directory records), and its own forwarder record.

- Query. This is a query from a client or directory server for a forwarder record, either explicitly (by name) or implicitly (by metadata or descriptor-derived data

field name	*format*
Complete Listing	`"GET /pan/ HTTP/1.1"`
Directory Burst	`"GET /pan/burst HTTP/1.1"`
Explicit Query	`"GET /desc/id/" FNAME SP "HTTP/1.1"`
Implicit Query	`"GET /desc/meta/" FMETA SP "HTTP/1.1"`
Publish Forwarder Record	`"POST /pan/ HTTP/1.1"`
Directory Update	`"POST /pan/directory-update HTTP/1.1"`

Table 4.3: Directory Request Formats.

field). See Section 4.1.4 for details.

- Publish Forwarder Record. This is an HTTP request from a forwarder to upload a complete forwarder record (possibly including an explicit forwarding path and metadata).

- Directory Update. This is an HTTP request from a neighboring directory server to upload status changes (deltas) since the most recent successful update.

4.2 Design Tradeoffs

The design of directory servers and propagation of routing information is more challenging in PAN than in BGP for several reasons:

- While BGP routing table entries consist of IPv4 prefixes, PAN routing table entries consist of *attribute sets*, a richer domain describing what can be reached using the network.

- PAN directory servers have the additional property that they provide information directly to clients as they construct source-routed circuits.

- While IPv4 prefixes are assigned by a central authority, there are no central authorities dictating the allocation of perspectives.

- Managing policies in PAN is more complex than in BGP. The PAN policy framework, described in Section 4.4, applies the techniques used to assign policy in BGP routing to this richer PAN domain.

The performance, scalability, and effectiveness of PAN largely derives from the design, implementation, and configuration of its directory servers. We consider the important issues in this section.

4.2.1 Structured versus Unstructured

Perhaps the first design question about our directory service is, considering the extensive research in distributed hash tables (DHT), whether we should implement our directory service using a structured network with $O(\log n)$ lookup operations rather than an unstructured network with fewer performance guarantees and more complexity.

There are several problems with using DHT, the most important of which for our purpose is the fact that DHTs assume a full mesh of connectivity. We want to allow an unstructured, organic growth of our network. Imagine, for example, using DHT to propagate BGP routing tables. This would be necessarily impossible, because the DHT itself requires some notion of connectivity that does not exist until the underlying network itself about which the DHT carries information is in place! Now, there are some potential solutions to adapt DHTs to work across multiple transport

domains (63), but these are naturally quite cumbersome, and to some extent they obviate the arguments that might otherwise make a DHT a good choice.

One of the key characteristics of DHT systems is the use of a uniform hash function to uniformly distribute load across servers, and the the hash function, which dictates which servers get which load, is essential to the the $O(logn)$ routing performance. However, the information that PAN stores is to a large extent location-dependent, and that location-dependence is, after all, the reason for our service. It would be detrimental to scalability and deleterious to server incentives to store information haphazardly throughout the network, when it makes more sense for individual directory servers to just store the information relevant to themselves.

Finally, DHT technology, as it exists today, has important security weaknesses. For example, to our knowledge there are no existing implementations of DHT that eliminate attacks related to influencing what a client thinks about which nodes are part of the network. There are also a plethora of theoretical attacks, including Sybil attacks, for which the proposed solutions are both cumbersome and unscalable (122; 36). These realities about security are among the primary reasons why Tor does not use DHT, despite the fact that Tor makes the assumption that all nodes are fully-connected (39).

4.2.2 Propagating Forwarder Information

Propagating the self-certifying name of each forwarder carries the advantage that clients may explicitly specify each forwarder individually. However, this advantage comes at a cost, since self-certifying names cannot be aggregated. The result is

that individual directory servers must contain at least the name of each forwarder in the entire network, so that they can appropriately respond to explicit queries requesting any individual forwarder. But, we can further relax the assumption that each directory server knows about each forwarder by allowing directory servers the option of propagating only metadata, rather than entire summary records. Naturally, metadata fields may contain the names of the forwarders themselves, but we rely upon the discretion of the individual directory servers to negotiate which information is propagated through the network. The effect of limited, policy-driven propagation may be that directory servers proximate to a given set of target forwarders may be configured to propagate their names and metadata while directory servers farther from the forwarders may be configured to propagate metadata only, in order to improve network scalability. The result would be that clients close to the forwarders may be able to make their selection with greater specificity.

Figure 4.8 depicts the propagation of reachability information from a small set of forwarders to the rest of the network. The circles illustrate how directory servers in different regions of the network may contain different information about particular forwarders. Directory servers nearest to the forwarders each contain all of the information needed by a client who desires to build a circuit to one of the forwarders in question. Directory servers somewhat farther from the forwarders may not have descriptors for the forwarders themselves, but they may possess *Directory Records* containing *Summary* attributes that provide enough information for clients to issue queries for the individual forwarders by name. However, directory servers in regions most distant from the forwarders may not have knowledge of the names of the indi-

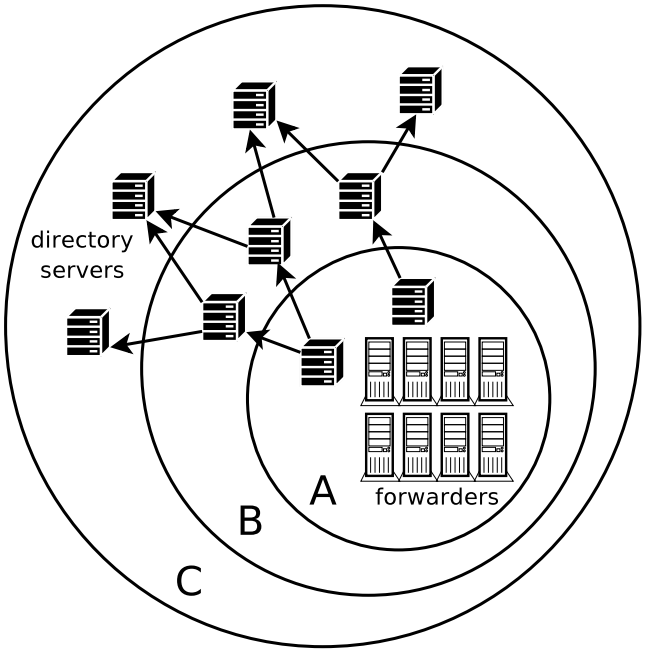


Figure 4.8: Metadata Propagation Regions.

vidual forwarders themselves, and may only have metadata describing the forwarders collectively. Clients using directory servers in these regions have no means of specifying those particular forwarders explicitly, but may only reach them in aggregate, by querying for attribute sets rather than explicit names.

Whether propagation of metadata is sufficient to assure reasonable scalability for PAN depends upon how PAN is used. For example, BGP scalability is limited by the number of independently propagated prefixes. Aggregation helps to some extent, since each prefix may correspond to thousands or even (theoretically) millions of individual hosts, but as we consider shorter prefixes, it becomes clear that at some level, the hierarchy ends, leaving each individual BGP listener with a table containing hundreds of thousands of distinct prefixes.

If the set of PAN forwarders were arranged such that there were exactly one per BGP prefix, with each forwarder as a directory server, and if peering relationships among directory servers topologically corresponded to peering relationships among autonomous systems, and if each client expected the ability to identify each PAN forwarder explicitly, then in theory the scalability of the Perspective Access Network would be essentially the same as that of the BGP network that exists today. However, this pattern of deployment and usage might not be what we can expect in a future PAN. Also, it is possible for multiple PANs to exist concurrently; private organizations might deploy their own PANs for their exclusive use.

For example, we might imagine that PAN would be used to link private networks, as we describe in Section 3.6, in which case we might assume that there would be one PAN forwarder in each private network. Since there are millions of private networks of this sort, an assumption that each would require the ability to identify each other explicitly could seriously constrain the scalability of PAN. However, we can resolve this by stipulating that clients who want to access specific destination forwarders know *a priori* how to reach directory servers that contain the necessary information for learning how to construct circuits that terminate at those specific forwarders. (Such instructions could be preconfigured in the PAN software at the time of distribution, for example.) PAN provides an architecture that allows communities of this sort to develop without overconstraining their structure.

Another use of PAN might be to have individual volunteers provide views of the world to be used at a high level of granularity; for example, clients might specify the names of particular countries or particular ISPs. In this situation, the exclusive

propagation of metadata might improve scalability considerably.

4.2.3 Responding to Queries

Suppose that a client issues a query for information that a particular directory server cannot provide but knows how to find. The directory server then has a choice. It may issue a *referral*, telling the client how to retrieve the information itself from other directory servers in the network, or it may treat the query *recursively*, forwarding the request on behalf of the client, and ultimately responding to the client with the information in the same manner that it would had it possessed the information at the time at which it received the query.

The difference between recursive and non-recursive (referral) responses to queries is comparable to the difference between their analogues in DNS. Referrals have the advantage that directory servers do less work, so servers under heavy load may wish to use this method. Recursive queries have the advantage that clients do less work and directory servers may cache the results. An enterprise may want to deploy servers that support recursive queries to allow clients to take advantage of requests made earlier by other clients if available, and possibly avoid some extra network traffic in the general case.

Refer to Figure 4.9. If a client queries a directory server that is configured to treat queries recursively, then the server manages the query on behalf of the client, and returns a response once it has received an adequate response from the directory server(s) that it uses to resolve the query.

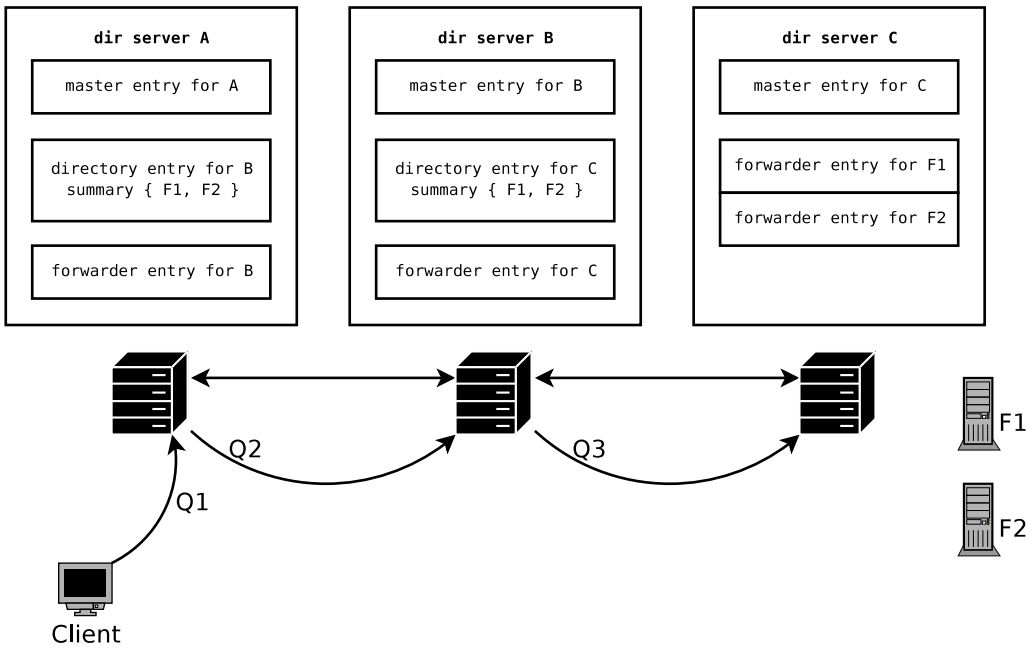


Figure 4.9: Recursive Queries.

4.2.4 Repeated Queries and Circuit Length

It is entirely conceivable that on occasion, two servers that are both generally accessible from the same set of clients possess different data, such that one server refers clients to the other. In such circumstances, we want a means by which the server can send a hint to the client suggesting that it should try connecting directly to the other server, so that it might avoid creating an unnecessarily long circuit for subsequent queries and the data plane. So, we have the directory server add a special attribute to its query response, indicating to the client that it **may** try querying the other directory server directly. Depending upon a variety of factors, the client may choose not to follow this recommendation, or the client may not be able to contact the other directory server directly. However, the practical utility of reducing circuit

length in circumstances like these is worth the added complexity.

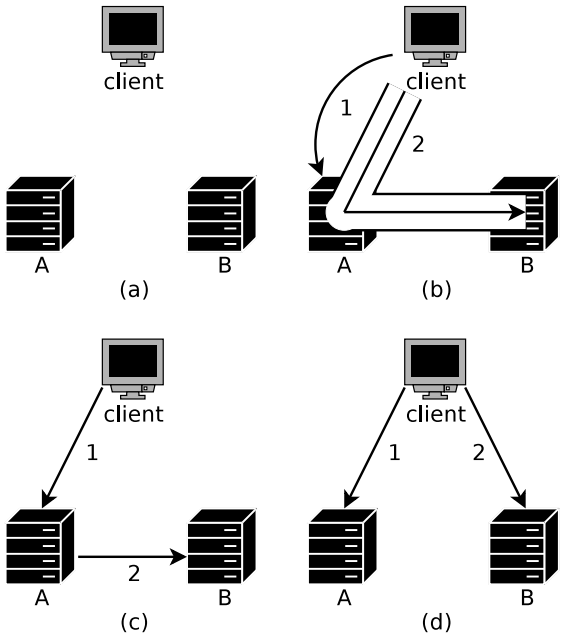


Figure 4.10: Handling Queries.

Figure 4.10 provides a summary of the ways in which a directory server may handle a query. Assume that a client requests a forwarder known only by the directory server labelled B, but it asks the directory server labelled A instead. Directory server A may handle the query in three distinct ways. In the first case, labeled (b), directory server A handles the query non-recursively, telling the client to build a circuit through A to B so that it can issue a query to B. In the second case, labeled (c), directory server A handles the query recursively, querying B for the requested information and responding to the client with the result. In the third case, labeled (d), directory server A has reason to believe that the client might be able to contact B directly, for example by observing that B and the client are on the same network. In this case, A

handles the query non-recursively but provides a hint to the client suggesting that it might be able to contact B directly, avoiding an unnecessary extension of the circuit to A. Of course, the client **may** disregard this hint, and the client should accept that the hint **may** be incorrect.

4.3 Configuration

A number of parameters govern how individual PAN directory servers interact with forwarders, clients, and their peers. These parameters include *peering directives* and *policy directives*. Peering directives, which we describe in this section, allow individual directory servers to make local decisions about what circuit-building information to propagate to their neighbors, or whether to summarize the information and require downstream clients to manage multiple queries. Policy directives, which we describe in Section 4.4, allow operators of directory servers to control aggregation, specify which information to propagate by attribute and propagation path, and manage network resources.

4.3.1 Filtering and Aggregation

Recall that directory servers have control over the contents of *Metadata* and *Compiled-Metadata* attributes. *Filtering* and *aggregation* configuration rules instruct directory servers how to adjust the values of these attributes. These rules are configured as part of the policy configuration described in Section 4.4. A brief overview follows; Section 5.3 provides a full treatment.

Directory servers may be configured to *filter* certain metadata. This may be

desired if a directory server chooses not to propagate certain kinds of perspective information to certain other directory servers.

Additionally, directory servers may be configured to aggregate metadata carrying perspective information, improving scalability. Two forms of aggregation are possible. The first form of aggregation involves collapsing substantively identical nodes (i.e., same attributes) into a single attribute set and advertising that attribute set. Since substantively identical nodes offer the same perspective as far as a client is concerned, no information is lost in this process. The second form of aggregation involves collapsing substantively similar, but not identical, nodes (i.e., partially matching attributes) into a more general attribute set by *single-attribute aggregation* (or by *subdivision*, as discussed in Section 5.3.1, which may be considered a special kind of aggregation but is not part of the policy configuration).

Information is lost as directory servers decide what information to discard (i.e., the extent of aggregation, subdivision, and filtering) to reduce the number of distinct sets of metadata to a reasonable value. The directory server **should** then set a flag indicating what data has been discarded, so that downstream directory servers can continue the same aggregation if they so choose, and so that clients have a hint about what upstream directory servers have answers to more specific queries.

4.3.2 Peering Arrangements

Directory servers establish manually-configured peering relationships with each other in a manner similar to how autonomous systems establish peering relationships with each other in a BGP context (note the security advantages afforded by having

humans explicitly configure the relationships). A special configuration file contains a list of neighboring directory servers (hereafter referred to as *neighbors*) along with peering policy and reachability information in the following format:

```
"neighbor" SP FNAME SP POLICY SP HOST ":" PORT          (4.3)
```

The `POLICY` field represents a peering directive that takes one of five values. Figure 4.12, located at the end of this chapter, provides a conceptual illustration.

- FULL. The directory server propagates both directory records and forwarder records received from the specified neighbor, adjusting the *Propagation-Path* attribute of each record by appending the name of the neighbor for loop-detection. The directory server **must not** adjust the *Forwarding-Path*, which contains the full source route. The directory server **may** alter *Metadata* and *Compiled-Metadata* attributes.

- PREPEND. The directory server propagates both directory records and forwarder records received from the specified neighbor, adjusting the propagation path by appending the name of the neighbor **and also** adjusting the *Forwarding-Path* of each forwarder record by prepending its own name. The difference between `prepend` and `full` is that `prepend` instructs clients to build circuits through this node en route to the destination, whereas `full` does not. Modification of other attributes is subject to the same conditions that apply to the `Full` directive.

- SUMMARIZE. The directory server propagates directory records received from

the specified neighbor, adjusting the propagation path by appending the name of the neighbor. However, rather than propagating all forwarder records from this neighbor, the directory server propagates only forwarder records corresponding to directory servers. In addition, the directory server creates a *Summary* attribute for this neighbor and adds the names of each forwarder whose forwarder record is received from this neighbor other than the neighbor itself. The directory server **should** also provide *one possible forwarding path* to each forwarder listed in the *Summary* attribute.

Similarly, the directory server creates a *Compiled-Metadata* attribute for this neighbor; filtering and aggregation rules, strategies for which are described in detail in Section 5.3, may constrain which entries are included. The directory server **may** define this attribute as the union of all *Metadata* attributes included in all forwarder records received via this neighbor except the forwarder record for the neighbor itself. The directory server **should** also provide *one possible forwarding path*, chosen according to local preference, to each *Metadata* attribute listed as part of the *Compiled-Metadata* attribute.

- PROXY. The directory server propagates neither directory records nor forwarder records received from the neighbor. Instead, the directory server creates a new directory record for this neighbor, according to the following specification. The *Summary* attribute of the new directory record **must** contain the full list (subject to filtering and aggregation rules) of all of the names of all of the forwarders for which the directory server received forwarder records from this neighbor **and** all of the names of all of the forwarders appearing in all *Summary*

attributes included in all directory records received from the specified neighbor. For each neighboring directory server N, the directory server **should** prepend its name and the names of forwarders along the path to N to all of the forwarding paths for all of the forwarders listed in the *Summary* attribute. Note that the result is *one possible forwarding path* to the desired forwarder; while clients **may** use this information to build a circuit, it is by no means canonical.

Similarly, the *Compiled-Metadata* attribute of the new directory record **should** contain the full list (subject to filtering and aggregation rules) of all *Metadata* attributes included in all forwarder records received via the specified neighbor **and** each element of each *Compiled-Metadata* attribute included in each directory record received via the specified neighbor. In this manner, clients **may** be referred to the specified neighbor when they request a forwarder name or attribute that propagated to this directory server via the specified neighbor. The directory server also retains *one possible forwarding path*, chosen according to its local preference, to each *Metadata* attribute listed as part of the *Compiled-Metadata* attribute. The difference between `proxy` and `summarize` is that `summarize` propagates directory records indiscriminately, whereas `proxy` propagates only one directory record for the given neighbor, accumulating all of the forwarders in all directory records received from that neighbor into its *Summary* attribute.

- NONE. The directory server does not propagate anything received from this peer. This peering directive specifies that a directory server **should** send periodic directory updates to this neighbor but **should not** make use of any

directory updates that it receives from this neighbor.

Consider the network topology and configuration shown in Figure 4.13, in which the peering directive for directory server E is specified by the corresponding row in Table 4.12. With the `full` peering directive, E propagates all of the records that it receives, including the summaries that it receives from A and B. With the `prepend` peering directive, E propagates all of the records that it receives, but also adds its own name to the (otherwise empty) forwarding paths for each of the records. With the `summarize` peering directive, E propagates the directory records for A and B as before, but rather than propagating a forwarder record for D_1, it lists D_1 in the *Summary* attribute of the directory record for D. With the `proxy` peering directive, E does not propagate the directory records for A and B, since they are not direct neighbors. Instead, E includes all forwarders advertised by C, including A, A_1, B, and B_1, in the *Summary* attribute for C and does not include directory records for A and B. For the `none` peering directive, E propagates no directory records for C or D.

If a directory server is configured such that the hostname field of some `neighbor` directive takes the form `HOST "." FNAME ".exit:" PORT`, then the directory server **should** wait for the specified neighbor to build a persistent circuit to the directory server before it attempts to establish contact (i.e., request a burst) with that neighbor.

Refer to Figure 4.13, located at the end of the chapter, for an example of how peering arrangements affect propagated records.

4.3.3 Propagation of Perspectives

There are two reasons why directory servers may choose not to propagate all data received from their neighbors to the rest of the network: scalability and policy. In the former case, directory servers may choose to aggregate data in order to reduce network load, and in the latter case, directory servers may choose to avoid telling their neighbors about received advertisements to conform to legal restrictions or to avoid receiving unwanted traffic in the data plane.

There are two design tradeoffs relevant to aggregation. First, there is a natural tradeoff between scalability and query *latency*. In small networks, it may be acceptable to propagate full information about all perspectives, including full paths and descriptors to any potential client location. The benefit is that clients experience improved performance since one query will be sufficient to provide a client with all of the information that it needs to construct a complete circuit. However, in large networks, maintaining this consistency throughout the entire network can be quite expensive in terms of directory server resources required, and arguably more expensive than the additional connection setup latency imposed by requiring either successive (non-recursive) or forwarded (recursive) queries.

Second, there is a natural tradeoff between scalability and query *specificity*. In small networks, it may be reasonable to provide a means by which all clients can specify each forwarder uniquely, but in large networks, the cost to scalability associated with maintaining specific information about each forwarder at each directory server in the entire network may be infeasible. PAN *policy directives*, described in Section 4.4, provide aggregation commands that allow individual directory servers to

make local decisions about how to aggregate individual perspectives.

If the metadata have been aggregated, and a client wants to request a more specific perspective than that afforded by the metadata possessed by the directory server that it chooses to query, then it **must** choose some matching subset of its desired query and query a directory server matching that subset. If the client does not succeed in finding a suitable perspective while following a certain path, then it may backtrack, but the client could potentially query an arbitrary number of directory servers before establishing with certainty that a perspective matching its query does not exist.

4.4 Policy Framework

The configuration of each directory server includes a *policy* that defines which routes to accept, which routes to propagate, and how to assign preferences among routes. We use the Routing Policy Specification Language (RPSL) as a starting point (4). By selecting the relevant features of RPSL, adapting them to handle the additional complexity associated with perspective descriptions, and adding some features to improve incentives for deployment, we create a Perspective Routing Policy Specification Language (PRPSL), a form of RPSL adapted for use with PAN directory servers.

We begin with the IETF-specified RFC describing RPSL. Recall from Chapter 3 the analogy between PAN directory servers and autonomous systems. We provide our specific changes below, followed by some illustrative examples.

4.4.1 Modifications to RPSL

Since the PAN directory servers effectively play both the role of router and the role of autonomous system in a BGP routing environment, we can eliminate all of the router-specific classes and attributes (e.g., the `inet-rtr` class). The classes for contact information (the `role` class) and extensibility (the `dictionary` class) may still be useful, but we do not propose any modifications or extensions.

The existing RPSL `as-set` and `peering-set` classes are both designed to refer to sets of autonomous systems, so we change the definitions to specify a set of directory server identifiers (all directory servers are forwarders, so these are the same as self-certifying forwarder identifiers, specified by `FNAME` as given in Section 4.1) instead. We also introduce a new class, `forward-set`, which specifies a set of forwarder identifiers (also represented by `FNAME`); the purpose is to disambiguate cases in which we refer to directory peers and cases in which we refer to individual forwarders.

The existing RPSL `route-set` class describes routes in terms of IPv4 prefixes; routes in PAN are described by perspectives instead. Hence, we change the definition of the `route-set` attribute as described in Section 3.3 (specifically Table 3.1).

Similarly, the existing RPSL `filter-set` class describes filtering rules in terms of autonomous systems and IPv4 prefixes; PAN filtering rules for perspectives are described in Section 3.3. However, we do not only want the ability to filter routes based upon perspectives; we also want the ability to filter routes based upon how perspectives were received as well. For example, a directory server may want to filter a route advertisement based upon whether it was received from some particular peer, whether some particular forwarder appears in its forwarding path, or whether some particular

directory server appears in its propagation path. Hence, our revised `filter-set` class incorporates the filtering rules described in Table 3.2, logical operators `AND`, `OR`, and `NOT`, as well as the following additional filtering options:

- `neighbor <as-set>.` Returns **true** if the neighbor from which a given route advertisement was received is a member of the list of directory servers given by `<as-set>`.
- `forwarding-path:<as-set>.` Returns **true** if any of the forwarders listed in the *forwarding path* attribute of the given route advertisement is a member of `<as-set>`.
- `propagation-path:<as-set>.` Returns **true** if any of the directory servers listed in the *propagation path* attribute of the given route advertisement is a member of `<as-set>`.

We simplify the `aut-num` class to eliminate the features irrelevant to PAN perspectives, redefining the attributes as follows. The `aut-num` attribute describes the identity of the directory server in question; its format is given as a directory server identifier. Table 4.4 provides the syntax for our modified `import` and `export` attributes (note the simplifications), as well as two new attributes, `expose` and `limit`. We define the attributes as follows:

- IMPORT. This attribute specifies that a route matching the indicated `filter` set should be accepted from the neighbors matching the indicated `peering` set. The optional `pref` argument indicates the relative preference to assign to routes accepted via this filtering rule.

attribute	*syntax*

`import:`	`from <peering-1> [pref <integer-1>]`
	`. . .`
	`from <peering-N> [pref <integer-N>]`
	`accept <filter>`

`export:`	`to <peering-1> [pref <integer-1>]`
	`[accounting <accounting-set-1>]`
	`. . .`
	`to <peering-N> [pref <integer-N>]`
	`[accounting <accounting-set-N>]`
	`announce <filter>`

`expose:`	`to <forward-1> [pref <integer-1>]`
	`[accounting <accounting-set-1>]`
	`. . .`
	`to <forward-N> [pref <integer-N>]`
	`[accounting <accounting-set-N>]`
	`announce <filter>`

`limit:`	`for <accounting-set-1> [allocate <bandwidth-1>]`
	`. . .`
	`for <accounting-set-N> [allocate <bandwidth-N>]`

Table 4.4: MODIFIED SYNTAX FOR RPSL `aut-num` CLASS ATTRIBUTES. *PAN simplifies the* `import` *and* `export` *class attributes but preserves the use of these attributes to assign preferences. The new* `expose` *attribute directs how directory servers may answer requests from clients. The new* `limit` *attribute and the associated* `accounting` *action govern the management of network resources.*

- EXPORT. This attribute specifies that a route matching the indicated `filter` set should be announced to the neighbors matching the indicated `peering` set. At most one route for a given perspective may be announced to each neighbor. The optional `pref` argument indicates the relative preference to assign to routes announced via this filtering rule; preferences specified via the `export` attribute have precedence over preferences specified via the `import` attribute for announcements to peers. The optional `accounting` action takes a single argument, an `accounting-set`, which corresponds to a particular category of

traffic whose combined bandwidth is limited using the `limit` attribute.

- EXPOSE. Unlike BGP speakers, which advertise routes only to their peers, PAN directory servers advertise routes to clients as well. The `expose` attribute allows directory server operators to specify which routes to advertise to clients when they request a route to a particular perspective. Clients optionally authenticate to the directory server, and routes matching the specified `filter` set are exposed to clients matching the specified `forward` set. The default behavior is to advertise all routes to any client that asks. The optional `pref` argument indicates the relative preference to assign to routes advertised via this filtering rule; preferences specified via the `expose` attribute have precedence over preferences specified via the `import` attribute for announcements to peers. The optional `accounting` action takes a single argument, an `accounting-set`, which corresponds to a particular category of traffic whose combined bandwidth is limited using the `limit` attribute.

- LIMIT. This attribute describes the volume of traffic that a directory server will carry for routes grouped in a particular `accounting-set`. The `accounting` action is useful as a means of limiting the volume of traffic that a directory server will handle for its neighbors, possibly to ensure that its implemented forwarding policy matches its incentives.

Note that both `pref` and `accounting` options are strictly internal; they are not advertised to neighbors or clients in any form.

For the `route` class, we make no changes, though we emphasize the particular attributes that we use for aggregation:

- `components <filter-set>`. The set of routes that form the aggregate.
- `aggr-bndry <as-set>`. The directory servers in the given `as-set` define the aggregation boundary beyond which only the aggregate route is exported.
- `aggr-mtd {inbound, outbound}`. This attribute indicates whether aggregation is performed when the route is received or when the route is advertised.
- `export-comps <filter-set>`. The unaggregated perspectives indicated by the given `filter-set` are advertised outside the aggregation boundary specified by the `aggr-bndry` attribute (note that this field provides exceptions to the aggregation boundary; the purpose is to satisfy external policy constraints).
- `inject <as-set>`. The directory servers in the given `as-set` perform the aggregation indicated by this `route` instance.
- `holes <filter-set>`. The perspectives indicated by `filter-set` constitute standing exceptions to the aggregation rule indicated by this `route` instance.

4.4.2 Examples

Next, we provide some examples to illustrate how the policy framework can be applied to express routine policy configurations. We consider the topology given by Figure 4.11.

EXAMPLE 1. *Import Policy.* `DS5` accepts all routes except routes to Taiwan from `DS3` and accepts routes to Taiwan from `DS1` and `DS2` only.

```
aut-num: DS5
import:  from DS3 accept NOT loc:Taiwan.*;
         from DS1 AND DS2 accept loc:Taiwan.*;
```

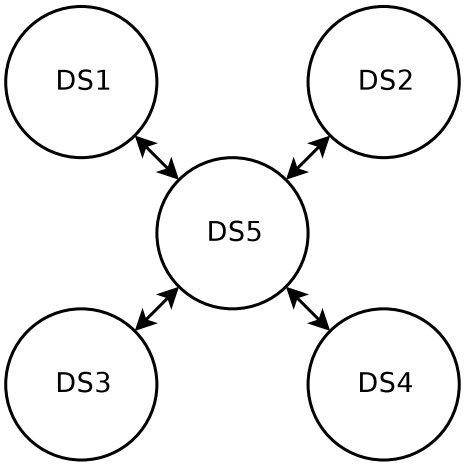


Figure 4.11: Example Topology to Illustrate Policy Configuration.

Example 2. *Preferences.* `DS5` accepts all routes from both `DS3` and `DS4`, but prefers routes from `DS4` (routes with a lower `pref` value are preferred). `DS5` also allows all clients to query for all perspectives.

```
aut-num: DS5
import:  from DS3 accept ANY pref 2;
         from DS4 accept ANY pref 1;
expose:  to F-ANY advertise ANY;
```

Example 3. *Exposure policy.* `DS5` accepts all routes from `DS3` and `DS4`, announces all routes to `DS1` and `DS3`, and announces all routes except routes to Tibet to `DS2`. `DS5` allows all clients to query for perspectives in `Massachusetts`, but allows only clients in filter set `FSET1` to query for perspectives in China that provide access to `political` content.

```
aut-num: DS5
import:  from DS3 accept ANY;
         from DS4 accept ANY;
expose:  to F-ANY advertise loc:US.MA;
         to FSET1 advertise loc:CN AND ief:+political;
```

```
export:  to DS1 AND DS3 announce ANY;
         to DS2 announce NOT loc:CN.Tibet.*;
```

EXAMPLE 4. *Provenance-specific filter sets.* DS5 accepts all routes from DS1, except those with F2 in the forwarding path, and all routes from DS2 except those with F3 in the propagation path. DS5 advertises all routes to DS3, except those with F3 in the propagation path, and all routes to DS4, except those with F2 in the forwarding path.

```
aut-num: DS5
import:  from DS1 accept NOT forwarding-path:F2;
         from DS2 accept NOT propagation-path:F3;
export:  to DS3 announce NOT propagation-path:F3;
         to DS4 announce NOT forwarding-path:F2;
```

EXAMPLE 5. *Accounting.* DS5 accepts all routes from each of its neighbors, but prefers them in a particular order; lower pref values indicate greater preference. However, when choosing which routes to advertise to DS4, DS5 prefers to advertise routes from DS3 rather than from DS2, when possible. Routes advertised to DS4 have a total combined bandwidth limitation of 50 MB/s, and routes advertised to DS3 have a total combined bandwidth limitation of 100 MB/s.

```
aut-num: DS5
import:  from DS1 accept ANY pref 1;
         from DS2 accept ANY pref 2;
         from DS3 accept ANY pref 3;
         from DS4 accept ANY pref 4;
export:  to DS3 announce ANY accounting ACCTSET1;
         to DS4 announce neighbor:DS2 pref 7
                accounting ACCTSET2;
         to DS4 announce neighbor:DS3 pref 5
                accounting ACCTSET2;
limit:   for ACCTSET1 allocate 100MB/s;
         for ACCTSET2 allocate 50MB/s;
```

EXAMPLE 6. *Aggregation.* If `DS5` receives advertisements from `DS4` for perspectives in Canada, it only propagates the advertisements as an aggregate outside the aggregation boundary indicated by the union of `DS5` and `DS4`. If `DS5` receives advertisements from `DS1` or `DS3` for perspectives that allow access to `news` content, then it aggregates the advertisements into a single advertisement for `news` content. Similarly, if `DS5` receives advertisements from `DS1` or `DS2` for perspectives that allow access to sites located in Iraq, then it aggregates the advertisements into a single advertisement for Iraq. Note that advertisements from `DS1` for perspectives in Iraq that allow access to `news` content are propagated as *two* aggregates: one for `news` and one for Iraq.

```
route:      loc:Canada;
components: loc:Canada.*;
aggr-bndry: DS4 OR DS5;
aggr-mtd:   outbound AS-ANY;

route:      ief:news;
components: ief:+news;
aggr-bndry: DS1 OR DS3 OR DS5;
aggr-mtd:   outbound AS-ANY;

route:      loc:Iraq;
components: loc:Iraq.*;
aggr-bndry: DS1 OR DS2 OR DS5;
aggr-mtd:   outbound AS-ANY;
```

Section 5.3 provides a discussion of strategies for aggregation and bandwidth accounting, highlighting their usefulness in achieving scalability and compatibility with operator incentives.

4.5 Dynamic Learning

The salient challenge in reconciling scalability and performance is to improve circuit building by reducing backtracking in a manner that does not require storing too much perspective information among the directory servers. There is a natural tradeoff between backtracking and advertising combinations of attributes, so directory servers should only advertise combinations of attributes based upon their usefulness to users, dynamically determining which queries are popular and which are not and deciding whether to advertise a perspective on this basis.

4.5.1 Exponential Problem in Managing Perspectives

Each perspective consists of a set of attributes, and users of a Perspective Access Network can use a set of attributes to specify a class of perspectives from which they would like to view the Internet. The cost of broadcasting all client requests to the entire network is overwhelming, so we propose a scheme by which perspectives advertise themselves to a distributed directory service. Unfortunately, if the number of possible attributes is too large, directory servers will be unable to store all possible combinations. For example, if there are thirty possible attributes, then there would be 2^{30} (over one billion) distinct combinations of attributes. By contrast, modern BGP routing tables contain fewer than two hundred thousand prefixes.

For some applications of Perspective Access Networks, clients would not be expected to request arbitrary combinations of attributes, and for those applications, scalability is not an issue. However, for applications that require more flexibility for clients, such as circumvention of political filtering, scalability becomes a challenge as

the distinctiveness of perspectives that clients request from the system may potentially grow quite large.

One possible way to address this problem is to advertise individual attributes separately, and require clients to systematically probe for combinations by guessing which directory servers know a particular combination and backtracking when they guess incorrectly. While this may work as a one-off solution, it is undesirable since it may take clients an unreasonably long time (e.g., the time required grows faster than linearly as the network size scales) to find a perspective. Fortunately, for some applications, only a subset of all possible sets of attributes are actually requested often enough by clients to be important to maintain in the directories. In these cases, we might only need assurance that directories will be able to efficiently accommodate the most commonly sought-after perspectives. Even if there is space to accommodate enough of the most commonly sought-after perspectives, the set of commonly sought-after perspectives may change over time, in which case we need a means of ensuring that we can handle the natural churn intrinsic to the set of popular perspectives.

We consider the *selection issues* that affect the tradeoff between expressivity of requests and performance of the directory servers:

- CHURN: the minute-to-minute popularity contest among perspectives, possibly resulting from governments adjusting their filtering policies.
- DRIFT: the change in the set of available attributes over time, possibly resulting from long-term social trends.
- REQUESTS: whether some sorts of requests are intrinsically more likely to occur than others, for example requests for combinations of a smaller number of

attributes.

With these issues in mind, we present a method for using hysteresis to improve stability, and we specify an algorithm for determining the most sought-after perspectives. Then, we describe a particular case study, political filtering, in which the number of potential perspectives scales exponentially with the number of distinct attributes, and for which our algorithm allows us to avoid excessive churn. Finally, we use arguments based upon evidence from the development of content-filtering techniques, legal discussion, and international policy literature to assert that the routing tables in the directory servers will be reasonably stable.

4.5.2 Hysteresis Approach

Suppose that N is the number of distinct attribute sets that are sufficiently popular to be meaningful, and M is the maximum table size that a directory server can handle. If N is smaller than M, then we have no problem. However, if N is larger than M, then churn will occur as the directory server discards existing attribute sets to make room for new ones that also meet the desired criteria. The instability could potentially affect downstream directory servers as well, as particular attribute combinations are repeatedly added and removed.

The solution in this case is to use hysteresis to induce stability among the attribute sets maintained in the routing tables of individual directory servers.

There are several established techniques for implementing hysteresis in lists, all of which have the effect of inhibiting changes in the set of entries. We outline some hysteresis techniques and the situations that brought them about:

BGP Route Flap Dampening

The goal of interdomain routing is to ensure that the various autonomous systems within the network have routes to all presently available destinations. If an autonomous system learns of a way (or a better way according to its own policy) to reach a particular prefix, it propagates the route advertisement to its peers. If an autonomous system discovers that a route to a particular prefix has become unavailable, it propagates a *withdrawal*, indicating that the route is no longer available. Occasionally, network errors or misconfiguration cause *route flapping*, a condition in which the route to a particular prefix is advertised and withdrawn repeatedly. Unstable routes cause problems by unnecessarily increasing the traffic between BGP peers and consequently increasing the processing performed by these peers. The task of avoiding route flapping presents a challenge: while accepting or maintaining an inferior route to a destination for a period of time is not a disaster, accepting or maintaining an invalid route to a destination can be a serious problem.

Route flap dampening is designed to reduce the impact of unstable routes on the network by systematically inhibiting the acceptance of new routes. There are two specific approaches. The first approach uses fixed timers to enqueue updates for a period of time and then accept all of the updates atomically, as a batch. Since route flapping tends to occur over intervals of tens of minutes to hours, the effective use of fixed timers alone to control dampening would necessarily have a significant deleterious impact on routing table convergence. The second approach is to suppress the acceptance of routes that have been recently observed to be "flapping." Thus, if a route is added and removed quickly, it receives a penalty that exponentially

decays over time, and if the accumulated penalty value is too high, then further advertisements of that route are ignored until the penalty value subsides to a level that permits the acceptance of the route (139).

Securities Indices

Securities indices exist to gauge the overall performance of a broad class of securities. Index funds are often created to allow investors to take positions on the broad class of securities without having to manually manage the weights of individual components themselves. Some investors use indices as a hedge, to take relative positions on individual securities relative to the market as a whole. Indices are periodically rebalanced to ensure that the set of securities remains an appropriate cross-section of the market. Since trading carries an associated cost, investors have an incentive to prefer investing in index funds whose composition remains stable.

There are several established ways of creating buffers that systemically limit the "turnover," i.e., the number of routine changes in the composition of the index. One method is to require certain performance metrics to be consistent for a few rebalancing periods before they cause a change in the index composition; consider the continued eligibility requirement for adjusted market capitalization of an individual index component of the NASDAQ-100 Index (92). A second method is to create a "buffer zone": suppose that there are two parameters for a single metric used to determine index membership. If the value of the metric is greater than the larger value, then the security is added to the index; if it is less than the smaller value, then the security is removed from the index; but if it is between the two values, the security

remains in its current state. This approach is described in the methodology overview for Morningstar Indices (89). A third method, which is particularly appropriate if the rules of the index require always having exactly N elements, is to maintain a ranking of the top $N + k$ elements, and only remove an element from the index if its rank falls below $N + k$. The NASDAQ-100 Index uses this strategy as well in its periodic Ranking Review (92).

Frequency-Based Replacement

Virtual memory caching reduces the incidence of high-latency disk access. Knowing which pages to cache and which to discard is the critical challenge, and the best decisions are those which accurately predict future page requests. Relatively naive strategies such as discarding and replacing pages that are least recently used (LRU) can help, but they can be suboptimal. A better approach might be to investigate the frequency of requests for individual pages, and not discard pages if they have been accessed frequently in the recent past.

Maintaining detailed records of the frequency of requests and the times at which they occurred is potentially cumbersome, requiring substantial time and space. One way of handling this issue is to use *generational replacement*, which involves organizing cached pages into tiers based upon the frequency of observed requests (62). After a page is initially requested, it spends some time cached in a pool for newly cached pages. Subsequently, rather than replacing this entry, it is moved to a pool in the middle of a priority chain. If a page is requested sufficiently frequently, it is promoted to a higher-ordered pool; if it is requested sufficiently infrequently, it is demoted to

a lower-ordered pool. When a new page is cached, the page to be replaced is chosen from the lowest-ordered non-empty pool. By providing some differentiation between pages that are naturally requested frequently over a long period of time and arbitrary pages that might happen to be accessed once, this strategy allows a smarter prediction of future page accesses in real-world scenarios.

Discussion

The directory service in Perspective Access Networks has some characteristics of a routing table and some characteristics of a fixed-size securities index, so the table of attribute sets reflects these characteristics. The table of attribute sets is like a fixed-size securities index in that directory servers want the number of entries in the table to remain bounded. Thus, for maintaining the set of popular queries, we use a stability buffer. In both cases, the directory servers want to minimize the number of unnecessary updates.

Additionally, the table is like a routing table in that individual directory servers must propagate new paths for the same attribute set if they are received and selected to maintain path vector correctness; hence, for routing table changes, we choose a method akin to route-flap dampening. Suppose that a given attribute set is listed among routes available to a particular directory server. If the route for that attribute set is repeatedly advertised and withdrawn, then that may result in propagating changes to neighbors (regardless of whether the attribute set is a combination of attributes or not). So, to reduce the effect of such oscillatory behavior, if we observe that a route is being repeatedly advertised and withdrawn, we suppress the propagation

of advertisements for that route for some time period.

We recommend the following approach. To determine the set of attributes, we use the $N+k$ approach: suppose that a directory server wants to advertise a table of size N. Then, the directory server will track the top $N+k$ queries for attribute sets for which a route is available. A new attribute set will be added only to fill a vacancy left by a departing attribute set, and an old attribute set will be removed only if its rank falls below $N+k$. We assume that the directory server has the space necessary to store entries for a substantially larger number of attribute sets for the purpose of counting frequencies. (Presumably, queries that occur fewer than some minimum number of times per unit time period can be discarded from this larger list.)

4.5.3 Algorithm

We provide pseudocode for the behavior of the directory server in response to two triggers related to perspective ranking: (a) the event in which a client request for a particular attribute set is received, and (b) a periodic event signalling the rebalancing of the list of the most popular attributes.

Suppose that the directory server is continuously maintaining a list of available routes to particular attribute sets. Also suppose that N and k are as described in Section 4.5.2, i.e., N is the number of combined attribute sets to advertise, and $N+k$ is the rank below which currently advertised combined attribute sets will be removed from the list. Then, our algorithm to determine which perspectives is as follows:

```
01 Routine Request( S ):
02     If NOT Count[S]:
03         Count[S] <- 0
04     Count[S] <- Count[S] + 1
```

```
05     ForEach S' in AllSubsets( S ):
06         Request( S' )
07     Return()
08
09 Routine Rebalance():
10     Ranked <- Sort( Count )
11     NewTable <- []
12     For Position := 0 to N + k :
13         If CurrentlyInTable( Ranked[Position] )
14             AND CurrentlyAvailable( Ranked[Position] ):
15                 Append( NewTable, Ranked[Position] )
16
17     ToAdd <- N - SizeOf( New Table )
18     For Position := 0 to N + k :
19         If CurrentlyAvailable( Ranked[Position] )
20             AND NOT CurrentlyInTable( Ranked[Position] ):
21                 ToAdd <- ToAdd - 1
22                 If ToAdd < 0 :
23                     Break
24                 Append( NewTable, Ranked[Position] )
25
26     Count <- {}
27     Return(NewTable)
```

The `Request` routine (Lines 1–7) runs whenever the directory server receives a request from a client for some attribute set `S`. The `Count` data structure is an associative array that maps attribute sets to positive integers, each of which represents the number of times in which the particular attribute set has been requested. The `AllSubsets` function returns all non-empty subsets of a given attribute set; notice that a request for an attribute set is a request for all non-empty subsets of itself as well.

The `Rebalance` routine (Lines 9–26) runs periodically, whenever the directory server wishes to update its list of popular attribute sets. The `Ranked` data structure is an array that holds the list of attribute sets requested since the last time `Rebalance`

was called, sorted by popularity The `CurrentlyInTable` function takes an attribute set and returns true if the attribute set is currently being advertised (i.e., it was determined to be in the top N last time), and false otherwise. The `CurrentlyAvailable` function takes an attribute set and returns true if there is an available route to the attribute set, and false otherwise. Lines 12–15 ensure that the top $N+k$ attribute sets remain in the list of attribute sets to continue propagating, so long as they are still available. Lines 17–24 fill the remainder of the list with the highest-ranked available attribute sets that were not among the top $N + k$.

In Section 5.5.1, we show how to evaluate our algorithm in the context of one particular use of Perspective Access Networks.

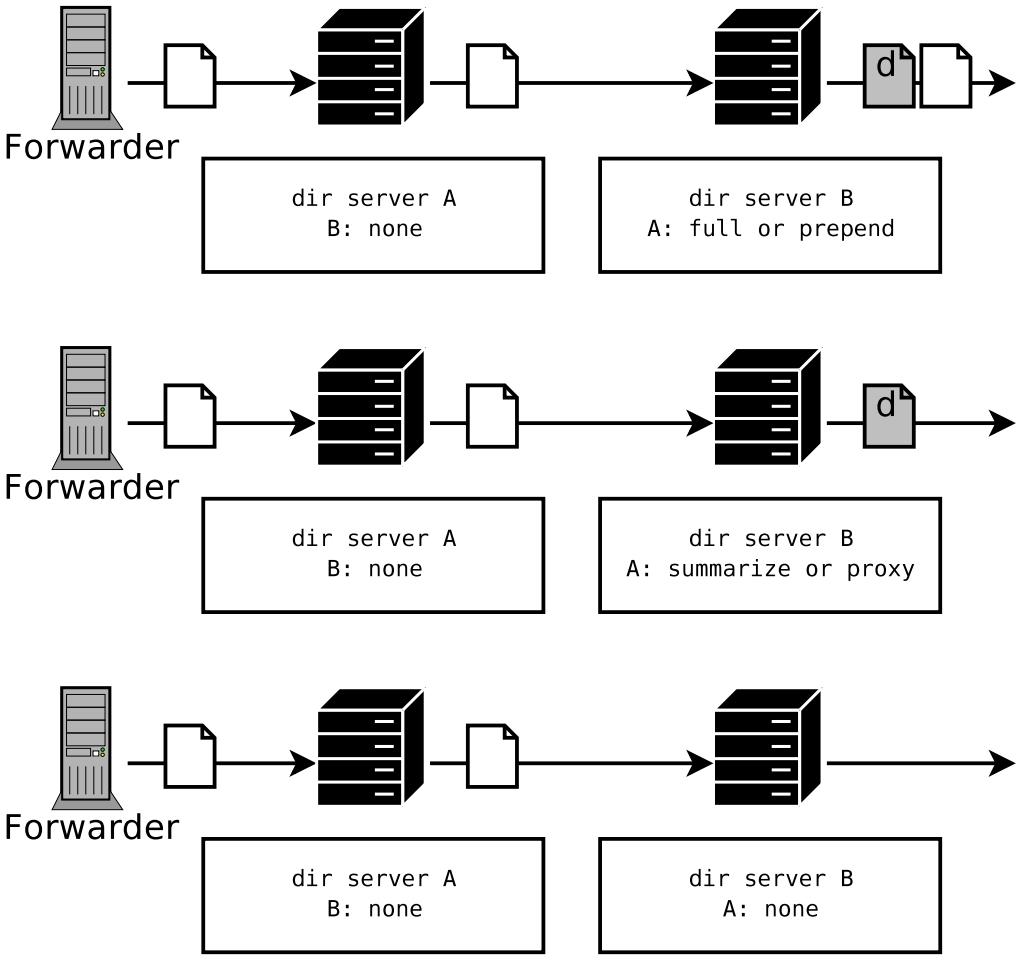


Figure 4.12: PEERING DIRECTIVES. *Suppose that a forwarder publishes to directory server A, and directory server B accepts updates from directory server A subject to some particular peering directive. If the peering directive is* FULL *or* PREPEND*, then B will propagate the forwarder record in addition to a directory record for A. If the peering directive is* SUMMARIZE *or* PROXY*, then B will include the name of the forwarder in the Summary attribute in the directory record for A. If the peering directive is* NONE*, then B will propagate no information about A or the forwarder records propagated from A. White pages are forwarder records; gray pages labelled* `d` *are directory updates.*

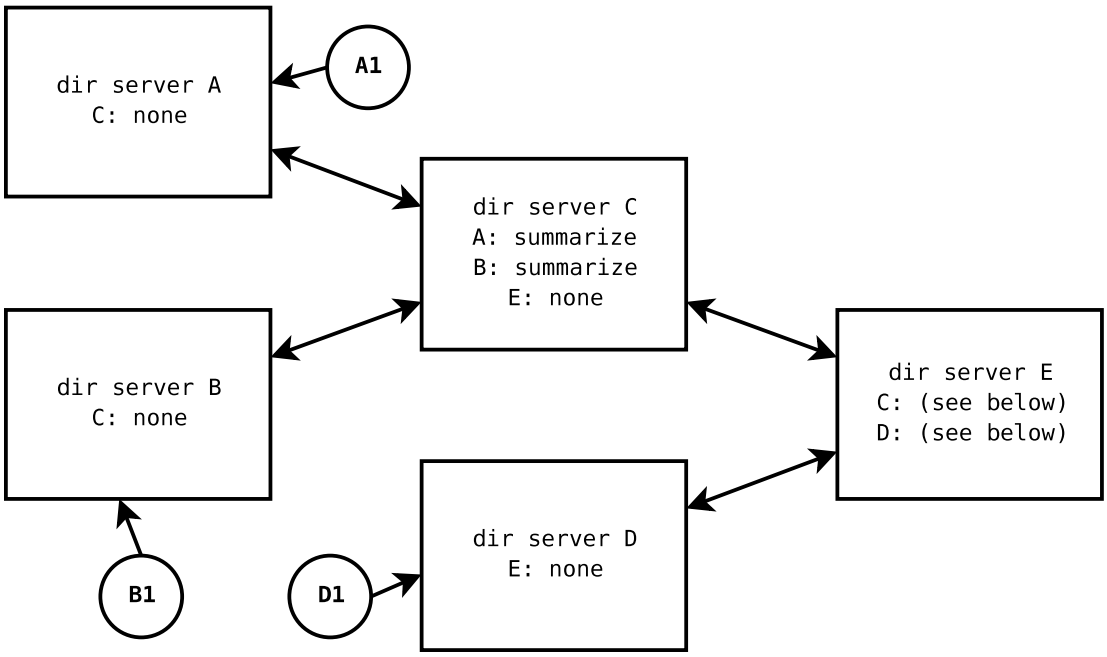


Figure 4.13: Peering Directives. *This example topology illustrates the functionality of the various peering directives. Refer to Table 4.5 for an explanation.*

directive	records propagated	attributes
full	dir A, fwd A	summary: A1
	dir B, fwd B	summary: B1
	dir C, fwd C	
	dir D, fwd D	
	fwd D1	
	fwd E	
prepend	dir A, fwd A	summary: A1
		fwd-path: E
	dir B, fwd B	summary: B1
		fwd-path: E
	dir C, fwd C	fwd-path: E
	dir D, fwd D	fwd-path: E
	fwd D1	fwd-path: E
	fwd E	
summarize	dir A, fwd A	summary: A1
	dir B, fwd B	summary: B1
	dir C, fwd C	
	dir D, fwd D	summary: D1
	fwd E	
proxy	dir C, fwd C	summary: A,A1,B,B1
	dir D, fwd D	summary: D1
	fwd E	
none	fwd E	

Table 4.5: Peering Directives. *Consider the scenario illustrated by Figure 4.13, in which* $\{A, B, C, D, E\}$ *are directory servers, with rectangular boxes indicating the peering directives for the indicated neighbors and* $\{A_1, B_1, D_1\}$ *are standalone forwarders. The table indicates what records are propagated and what corresponding attributes are defined when* E *applies the indicated peering directives for its neighbors,* C *and* D*.*

Chapter 5

Evaluation

In this chapter we provide quantitative and qualitative arguments to assess the technical merits of Perspective Access Networks.

Our quantitative results are based upon experiments performed on our realized prototype, Blossom. The implementation of Blossom is mostly intended to provide a proof of concept and a framework for discussing the technical issues. Nonetheless, we are able to empirically observe some of the more interesting issues related to scalability and performance.

Without a realistic user base in the live Internet, it is difficult to quantitatively determine the relative importance of the various design tradeoffs discussed in the previous chapters. However, the opportunities for qualitative evaluation are substantial. Part of this chapter is devoted to exploring the practical issues associated with deployment of Perspective Access Networks. We intend our judgments to provide some direction and insight for how to best capitalize on the features provided by our infrastructure.

We divide this chapter into four sections. The first two sections provide empirical analysis and discussion. In the first section, we assess the user experience by evaluating the performance of the Blossom client. In the second section, we consider the directory service; we explore the effects of the various directory server configuration parameters on network scalability. We also briefly discuss some issues related to the dynamic behavior of the PAN infrastructure. The last two sections provide qualitative evaluation and judgments about the practical usefulness of our system as proposed. In the third section, we focus upon deployability issues. Here we present some strategies for the use of aggregation to address network scalability as well as strategies for the use of filtering and bandwidth provisioning to address the incentives of PAN infrastructure providers. In the final section, we discuss the practical applicability of PAN, including what we see as its uses in the near-term and speculation about how PAN might be used in the future.

5.1 Client Performance

The experience of end-users of the PAN infrastructure is largely determined by the behavior of the PAN client. There are different components to client performance. The first component is the lookup process, which contributes to circuit setup latency; this factor is influenced by the degree of aggregation and the extent to which directory servers are able to answer queries from clients. The second component is the ongoing performance of the tunnel after it has been constructed. Recall that we intend PAN to be suitable for low-latency applications, such as Web browsing and interactive sessions; we evaluate the performance of our Blossom client with this decision in

mind:

- First, we assess circuit setup performance in detail, including both the selection of circuits using the Blossom directory servers as well as the construction of circuits using Tor. We show how to isolate the aspects of the observed performance that we can improve from the aspects that are dictated by the underlying network.

- Second, we briefly highlight relevant aspects of the performance of Tor, which serves as the underlying transport and circuit-building substrate for Blossom.

5.1.1 Circuit Setup

We deployed Blossom on about 300 PlanetLab nodes for the purposes of conducting empirical tests. To test setup latency for circuits involving multiple hops through the forwarding network and the effect of client queries on path setup time, we generated some paths of various lengths using randomly chosen PlanetLab nodes and constructed circuits using those paths. Using these paths, we performed two experiments:

- Generic Circuit-Building Test. We tested the time taken for Tor to build a circuit for a specified path by requesting to send TCP traffic to some port on the final node in the circuit. The results of this test are represented as solid triangles in Figure 5.2. Each triangle represents the median observed TCP connection setup latency using predetermined circuits of that particular length over ten independent trials. We are interested in using PAN for interactive

applications, and by comparison, the International Telecommunications Union recommends an average call setup delay of eight seconds for international calls via the ISDN (67). Furthermore, user studies have shown that users sometimes shift the focus of their attention after as little as two seconds (112).

- CIRCUIT-BUILDING TEST WITH QUERIES. In our second experiment, we tested the time taken for Tor to build a circuit according to a path that the Blossom client determines by iteratively issuing queries to each successive directory server along the path to the final node in the circuit. The results of this test are represented as hollow circles in Figure 5.2. Each circle represents the median observed TCP connection setup latency using dynamically determined circuits of that particular length over ten independent trials. In each case, the number of queries performed is equal to the number of hops minus one. Note that connection setup consistently takes longer when the Blossom client performs queries.

Whether a client will have to perform queries or not depends upon how directory servers within the Blossom network are configured. Figure 5.1 illustrates the interaction that takes place between a Blossom client and directory servers when the client extends the circuit from length n to length $n + 1$. The top portion of the interaction, marked "Query component," only occurs when the client issues a query before extending the circuit.

In both cases, since the process of extending a circuit from length n to length $n + 1$ involves sending messages back and forth over the entire $O(n)$ length of the circuit, the circuit setup time scales quadratically with the length of the circuit. The

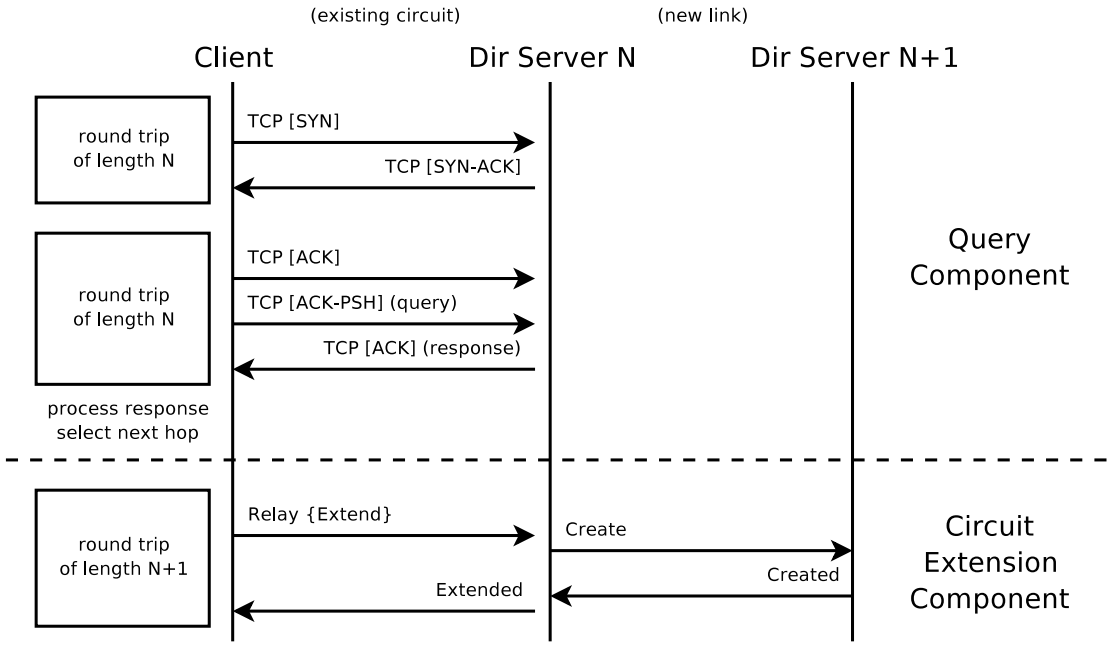


Figure 5.1: Extending a Circuit (with Querying). *Clients that do not already know the next hop in the circuit must first send a query to the current directory server before instructing Tor to extend the circuit.*

two parabolic lines in Figure 5.2 correspond to a quadratic least-squares regression of the data from each of the two experiments, respectively; Table 5.1 provides the coefficients. Observe that queries introduce noticeable additional latency, particularly as circuit length increases. Figure 5.3 presents the same results, except subtracting the expected network delay times between pairs of nodes (i.e., all of the round-trip times indicated in Figure 5.1). We obtained the pairwise network latency values from a set of measurements conducted by C. Yoshikawa.[1]

The circuit-setup experiments involved randomly-chosen PlanetLab (66) nodes. As a result, while the experiments do not correspond to worst-case scenarios, the results are "conservative" in the sense that the neighboring nodes are chosen without

[1]PlanetLab: All Sites Pings, `http://ping.ececs.uc.edu/ping/`

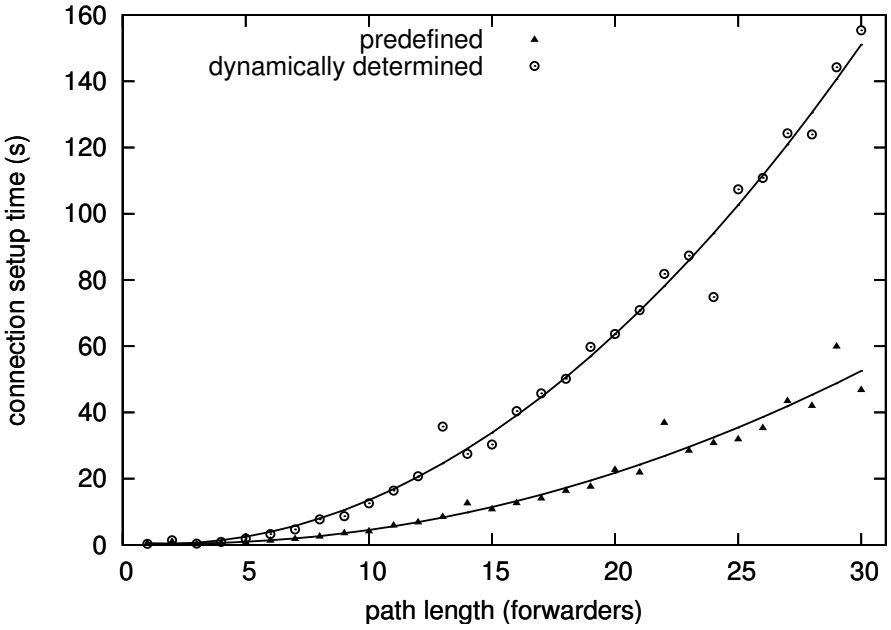


Figure 5.2: Circuit Setup Latency. *Time taken to build a circuit and establish an end-to-end TCP session for circuits of varying lengths. Circuits built according to predetermined paths are shown as filled triangles; circuits built via paths determined dynamically via Blossom querying are shown as hollow circles. The solid lines represent quadratic least-squares regression curves for the two experiments.*

regard for the underlying network topology. We suspect that in actual PAN networks, administrators of PAN directory servers would arrange themselves in a less random, more advantageous topology. Observe that network latency accounts for the vast majority of delay associated with connection setup. Unfortunately, there is no way to reimplement Blossom that avoids this delay; the only solution is to improve the underlying network. However, Figure 5.3 shows that system-internal delay accounts for some portion of the time spent during circuit setup, and this particular delay can potentially be improved by reimplementing Blossom. Note that this delay will also increase with circuit length, since establishing longer circuits involves interaction with a greater number of directory servers, which scales linearly with circuit length, and

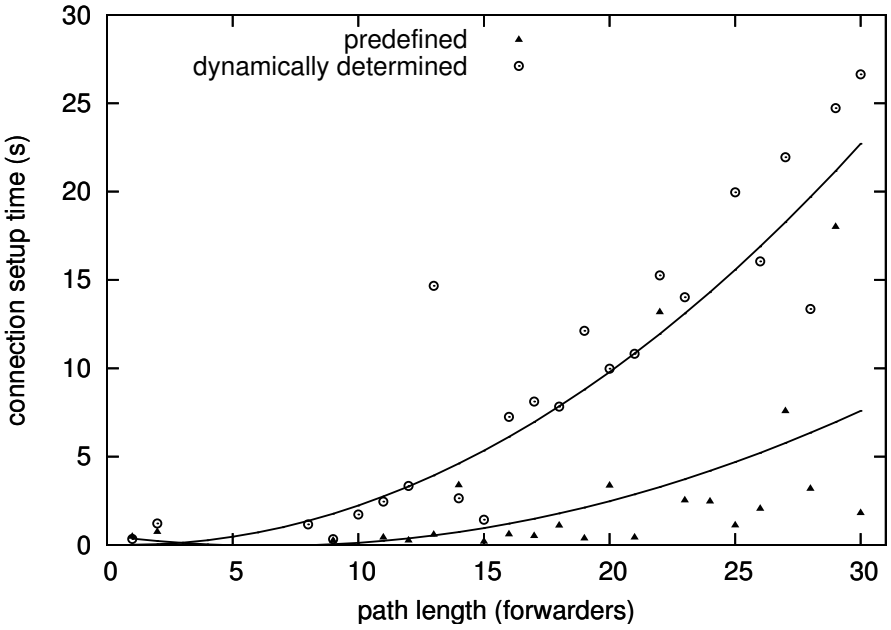


Figure 5.3: Circuit Setup Latency, Adjusted for Network Delay. *This graph presents the same experiments as Figure 5.2, but adjusted to remove the round-trip times introduced by network delay. Note that the scale of the y-axis differs from Figure 5.2.*

more cryptographic operations, which scale quadratically with circuit length.

As described in Section 4.1.2, when forwarder records providing access to the desired perspective do not exist, PAN clients **may** build a path based upon forwarder-specific or directory-specific forwarding path information contained in the *Summary* or *Compiled-Metadata* fields. Our experiments expose the following tradeoff: if a client tries to explicitly build a path based upon forwarding path information, it sacrifices some measure of control over the path as well as some confidence that the forwarding path information is accurate, but the process of querying all directory servers along the forwarding path degrades circuit setup performance.

experiment	*a*	*b*	*c*
predefined	0.0674	-0.2960	0.7310
dynamically determined	0.1867	-0.5926	0.7721
predefined minus RTT	0.0138	-0.1785	0.5263
dynamically determined minus RTT	0.0268	-0.0483	0.0330

Table 5.1: Coefficients for Quadratic Least-Squares Regression. *These coefficients define the parabolas defined by the equation* $ax^2 + bx + c = 0$ *for the experimental results illustrated in Figures 5.2 and 5.3.*

Overall, if we accept the ITU eight-second call setup delay recommendation for the PAN circuit construction process, our experiments illustrate that for sufficiently short circuit lengths (up to eight hops for dynamically determined circuits, up to twelve hops for predefined circuits), circuit setup latency is reasonable for human users.

5.1.2 Data Plane

Next, we consider the ongoing performance of circuits once they have been established. Tor provides a proof-of-concept of a special-purpose overlay network that routes general-purpose traffic, and the Tor experiment has demonstrated the successful, unmediated deployment of networks of this type for altruistic purposes. Blossom uses Tor (38) for building circuits and subsequently transporting the data of TCP streams between client applications and servers via the established circuits. Therefore, a thorough evaluation of Blossom thus includes a consideration of the appropriateness of using the data plane that Tor provides.

Presently, the Tor network is optimized for interactive applications, and empirically observed usage patterns reflect this fact. Researchers at Rice University have

discovered that the most popular uses of Tor are low-latency applications such as Web browsing (69). Traffic to the TCP ports most commonly used for web servers (80 and 443) constitutes over three-fourths of the traffic, and much of the remainder of the traffic appears to consist of low-latency instant messaging protocols such as IRC and interactive shell applications such as SSH. Other anonymity systems such as I2P[2] may be more well-suited to high-latency applications such as filesharing; the research to demonstrate this is currently inconclusive.

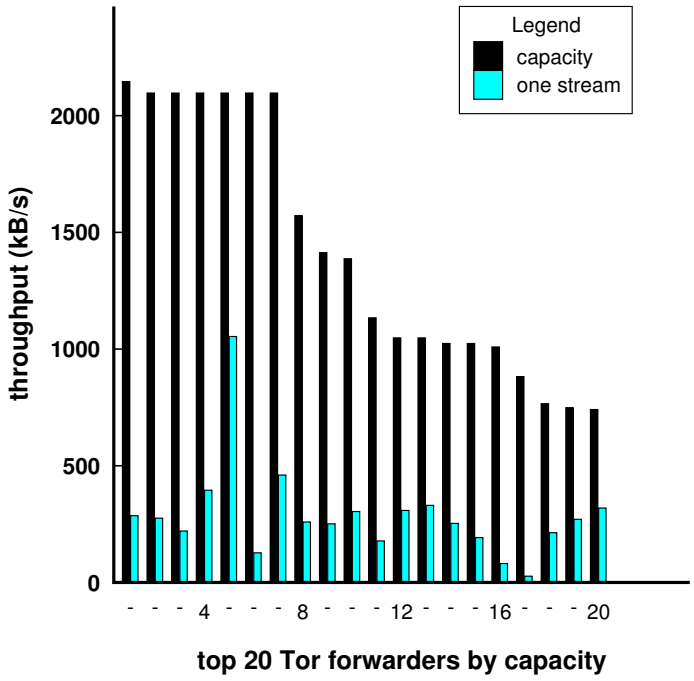


Figure 5.4: Throughput of High-Capacity Tor Nodes. *Data from 28 April 2006.*

The limitations of the Tor anonymity network as it exists today can be suffi-

[2]I2P, `http://www.i2p.net/`

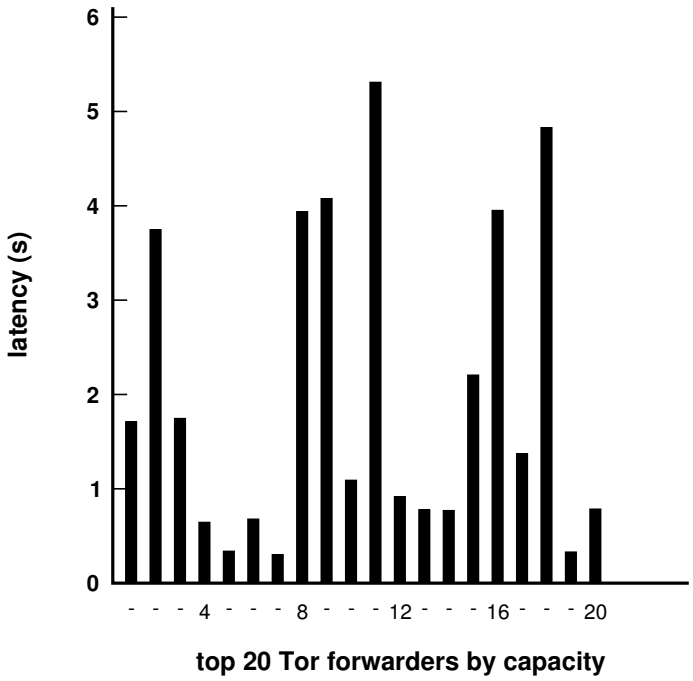


Figure 5.5: Circuit Setup Latency of High-Capacity Tor Nodes. *Data from 28 April 2006.*

ciently explained by the nature of the individual forwarders of which it is composed. Figures 5.4 and 5.5 show the throughput and latency of each of the twenty highest-capacity Tor nodes that exit to port 80, respectively.[3] (Note that these observations involve Tor exclusively and do not measure Blossom.) The most active Tor forwarders carry a peak bandwidth of over two megabytes of traffic per second. The current usage pattern indicates that a typical stream using one of these forwarders chosen at random can expect a sustained throughput in excess of 200 kB/s. Observe that the circuit setup latency for the Tor nodes is somewhat greater than the latency observed

[3]Capacity is determined by highest bandwidth achieved over a ten-second interval during the last 24-hour period. Refer to `http://tor.eff.org/cvs/tor/doc/tor-spec.txt` for details.

in our experiments described in Section 5.1.1; this may be the result of limitations related to the large number of concurrent connections among Tor nodes. As of April 2006, the high-performance Tor forwarders running at Harvard can expect to have established roughly 2000 concurrent connections at any given time.

Ultimately, the Tor network as currently implemented has some important shortcomings. In particular, Tor is not implemented in hardware, so individual Tor forwarders are not nearly as powerful as they could be. Additionally, the Tor network consists of low-cost, general-purpose personal computers operated by volunteers, largely on networks not designed to carry Tor traffic. Furthermore, the most recent implementation of Tor has serious performance limitations on popular operating systems, specifically Windows XP and derivatives. As a result, many of the servers are not performing close to their theoretical potential, and deployment has been somewhat hindered.

Additionally, there are some concerns about impact of the use of Tor circuits on end-to-end performance; two main factors affect performance. First, the core of the Internet generally does not constrain the bandwidth available to an end-to-end connection, and latency in the core is relatively small. However, the forwarders in the Tor overlay are generally personal computers and servers at the edge; if all forwarders and the client have similar, symmetric upstream connectivity, then a circuit of length n can be expected to increase perceived latency by a factor of $2n+1$. Second, the act of concatenating TCP sessions may interfere with TCP congestion control, causing degraded performance.

However, the popularity and usage patterns of Tor corroborate its utility for low-

latency applications. As this discussion has shown, performance of these systems is adequate for a variety of conventional Internet applications, and the choice of Tor for the Blossom data plane is therefore appropriate.

5.2 Directory Performance

To illustrate some of the design tradeoffs inherent to the PAN directory service, we performed empirical measurements using a deployment of roughly 300 nodes on PlanetLab. In our experiments, each of the nodes serves as a forwarder in the PAN overlay, and some subset of the nodes also serve as directory servers. We refer to nodes that perform just forwarding as *standalone forwarders*.

For each of our experiments, we assigned forwarders and directory servers at random from the set of PlanetLab nodes that we had previously determined to be responsive. As with the circuit-setup experiments described in Section 5.1.1, the selection process for these experiments assigns forwarder roles randomly, so the topologies that we chose are "conservative" in the sense that pairs of nodes that directly communicate with each other are determined without regard to the underlying network infrastructure. We suspect that pairwise communicators in most PAN networks deployed in practice would be (at least somewhat) topologically close rather than entirely random.

5.2.1 Infrastructure Performance

The PAN control plane consists of communication among directory servers and between individual directory servers and forwarders. To evaluate the control plane in terms of control messages and performance between directory servers, we generated a

symbol	description
N	number of nodes ($\sim$ 300)
n_f	number of standalone forwarders per directory server ($\sim$ 16)
n_d	number of directory servers ($\sim$ 20)
s_d	size of directory record (varies)
$\hat{s}_d$	size of forwarder record with summary (varies)
s_f	size of forwarder record ($\sim$ 4 kB)
δ	number of neighbors per directory server ($\sim$ 4)
T_d	directory update interval ($\sim$ 60 s)
T_f	forwarder fetch interval ($\sim$ 600 s)
T_e	forwarder record expiration ($\sim$ 86400 s)

Table 5.2: Control Plane Traffic Parameters.

number of different topologies by varying the topology, update frequency, and peering directives (as described in Section 4.3). Table 5.2 provides a list of the parameters relevant to our infrastructure experiments.

We then performed a series of experiments that involve selecting different combinations of values for T_d, n_f, and δ, as well as different peering directives (specifically, `full` versus `summarize` versus `proxy`). We observed the size and frequency of messages sent between directories and standalone forwarders as well as messages sent among directory servers. In practice, we expect low churn for perspectives, as we describe in Section 5.5.

In each case, we used a set of N nodes, selecting n_f standalone forwarders per directory server, leaving $n_d = \lceil N/n_f \rceil$. We organized the standalone forwarders into n_d groups of n_f, such that each forwarder in a group publishes its forwarder record to the same directory server and each directory server receives forwarder records from a fixed number of neighbors, as shown in Figure 5.6. Note that as we increase the value

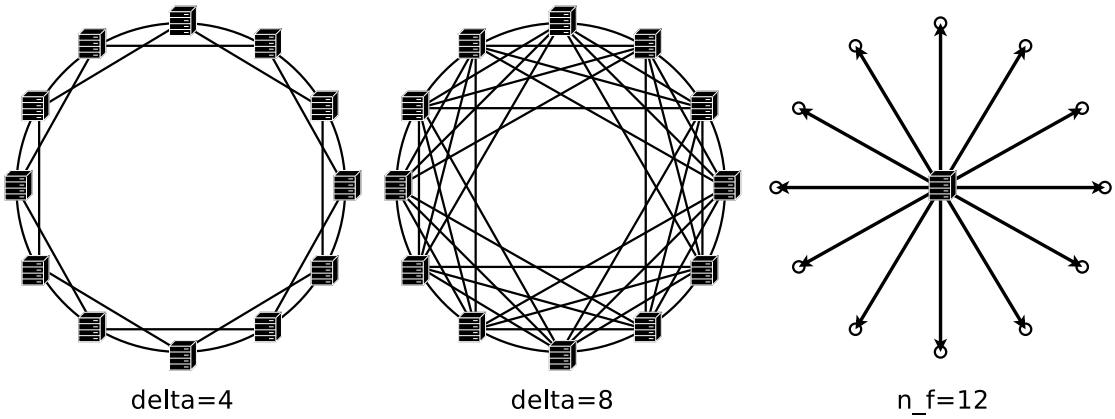


Figure 5.6: DIRECTORY TOPOLOGY. *In our experiments, we organize the directory servers in a symmetric, circular topology in which all directory servers have the same number of neighbors (δ) and the same number of standalone forwarders per directory server (n_f).*

of n_f, the number of directory servers decreases, since N is presumed to be constant.

For each experiment, we organized the set of the directory servers into a symmetric, circular topology in which each directory servers has exactly δ neighbors. Forwarders contact their assigned directory servers to publish their forwarder records and download the latest version of the directory every T_f seconds If the directory updates are reliable, then T_f depends entirely upon churn, and since we expect low churn, the value for T_f should not be too small. Directory servers push updates (such as changes to descriptors, withdrawals for forwarders that have failed) to other directory servers every T_d seconds.

Our experiments investigate the following questions:

- What effect does the degree of connectivity, δ, have on the overall amount of traffic on the control plane?

- What effect does the extent of clustering n_f have on the throughput of con-

trol messages sent amongst directory servers and between directory servers and standalone forwarders?

- What effect do peering directives `summarize` and `proxy` have on the overall throughput of control messages?
- What effect does the interval T_d between directory updates have on the transfer rate of control messages between directory servers?

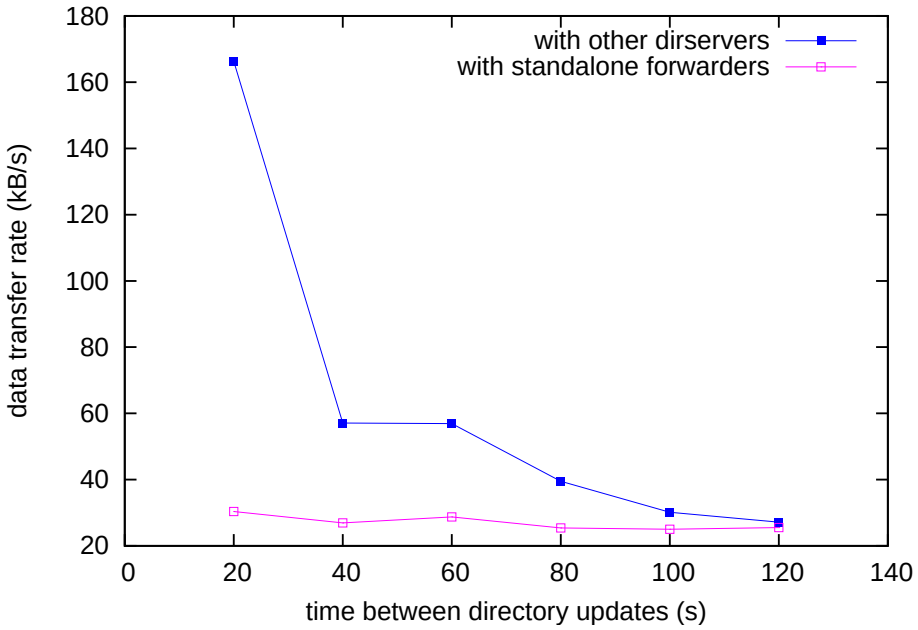


Figure 5.7: Directory Update Interval. *Effect of perturbing T_d while setting $\delta = 4$, $n_f = 8$, and peering directive* `summarize`. *(The data transfer rate shown is for the control plane.)*

By our model, the following equation governs the control data rate r generated by each directory server in the control plane, measured in bytes per second:

$$r = \frac{n_f u}{T_f} + \frac{\delta u}{T_k} \tag{5.1}$$

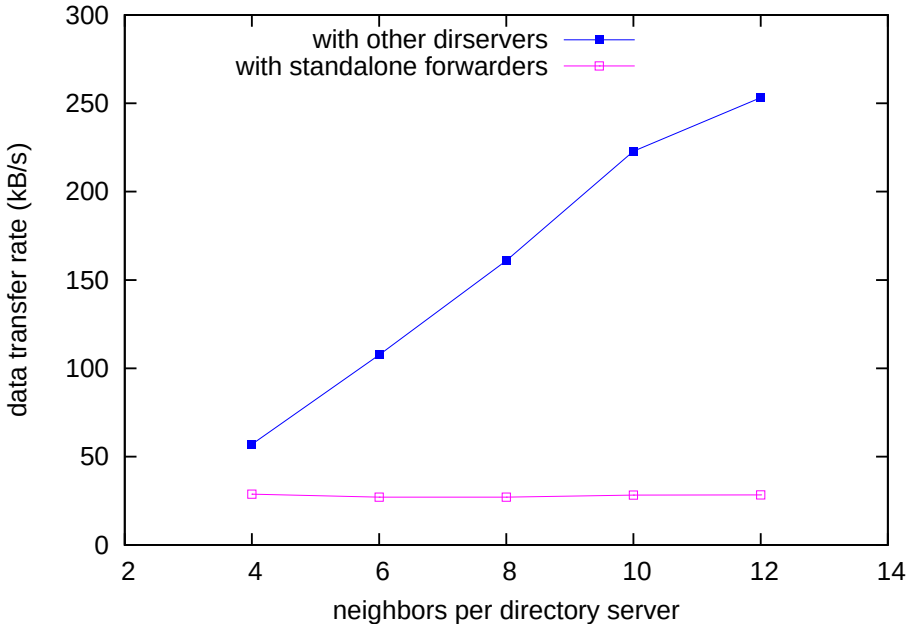


Figure 5.8: FORWARDER CONNECTEDNESS. *Effect of perturbing δ while setting $T_d = 60$, $n_f = 8$, and peering directive* `summarize`. *(The data transfer rate shown is for the control plane.)*

The first term describes the interaction with standalone forwarders, and the second term describes the interaction with neighboring directory servers. The value of T_k is determined by the extent to which the records held by individual directory servers have converged. In an ideal situation, the denominator of the first term would be exactly T_e, though our implementation makes no effort to achieve this goal. The relationships between the various interval values are given by the following expression:

$$T_d \leq T_k \leq T_f \leq T_e \tag{5.2}$$

The value of u in Equation 5.1 is determined by the particular peering directive used, and we use the following equations to model how u varies with topology and the size of records:

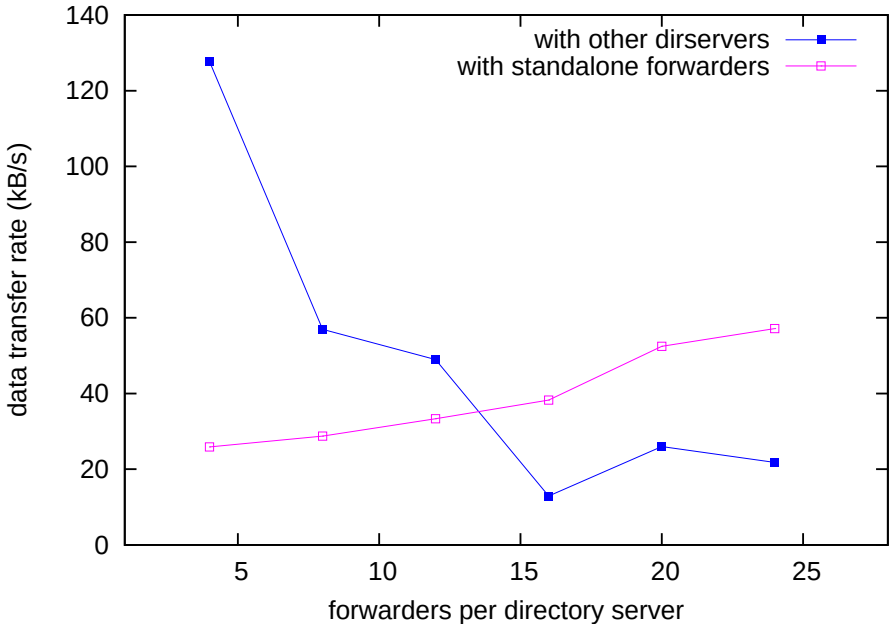


Figure 5.9: FORWARDERS PER DIRECTORY SERVER. *Effect of perturbing n_f while setting $T_d = 60$, $\delta = 4$, and peering directive* `summarize`. *(The data transfer rate shown is for the control plane, and the x axis represents δ, the number of directory server neighbors per directory server.)*

$$u_{\texttt{full}} = n_d s_d + N s_f \tag{5.3}$$

$$u_{\texttt{summarize}} = n_d \hat{s}_d + (n_d + n_f) s_f \tag{5.4}$$

$$u_{\texttt{proxy}} = \delta \hat{s}_d + n_f s_f \tag{5.5}$$

Figures 5.7 through 5.9 depict the approximate outbound data rate for individual directory servers as observed. The two terms in Equation 5.1 refer to the two lines in the figures.

Figure 5.7 illustrates the effect of varying the frequency of updates between directory servers. As the duration between updates increases, the quantity of outbound

traffic to other directory servers decreases in inverse proportion to T_d. So, improving the convergence time for the PAN routing tables requires a concomitant investment of bandwidth.

Figure 5.8 illustrates the effect of varying the number of directory server neighbors (δ) to which each directory server is connected. As δ increases, the volume of outbound traffic to other directory servers increases linearly, since changes in internal state are propagated to all directory server neighbors (because of stability, this is a potentially rare event). Therefore, improving the robustness of the system by increasing the connectivity between nodes also requires an investment in bandwidth. The figure shows the outcome of an experiment using the `summarize` peering directive, but it is important to note that if the `proxy` peering directive were used instead, then the volume of control plane messages would still increase proportionally with δ, but the size and frequency of the messages would be reduced, since each directory server is expected to share only δ (rather than n_d) directory records with each of its neighbors.

When we refer to "standalone forwarders," we could mean individual forwarders or collections of forwarders with the same perspectives (the `perspective` peering directive could be chosen to cause the second case to be treated as the first). However, our experiments do not take into account aggregation of forwarders with similar perspectives.

Figure 5.9 illustrates the effect of varying the number of standalone forwarders that publish their forwarder records to a given directory server. Since our experiments use a constant number of nodes, adjusting this parameter changes the ratio of directory

servers to standalone forwarders. Specifically, n_f increases while n_d decreases. Since we are using the `summarize` peering directive, the volume of traffic between a given directory server and standalone forwarders increases because n_f dominates the first term of Equation 5.1, whereas the volume of traffic sent to other directory servers decreases because n_d and n_f dominate the second term of Equation 5.1. So, increasing the number of "leaves" in the topology by decreasing the ratio of directory servers to standalone forwarders alleviates some of the traffic in the core of the network but increases traffic at the edges. Robustness is not necessarily affected, since forwarders can publish their forwarder records to multiple directory servers. While we do not show experimental results for that situation, we assert that directing each standalone forwarder to publish its forwarder record to m directory servers involves substituting mn_f for n_f in Equations 5.1 through 5.5.

5.2.2 Traffic Profiles

Figures 5.10 and 5.11 depict the average outbound control plane traffic volume per minute for a typical directory server. Figure 5.10 presents the outbound traffic between a directory server and its neighbors, given peering rule `summarize` and two different values of n_f. Observe that the traffic volume levels off after increasing for the first twenty minutes while PlanetLab nodes come online[4] and routing information converges. Figure 5.11 shows the average outbound control plane traffic volume per minute to standalone forwarders. The periodicity is the result of periodic directory fetches at time interval T_f on the part of standalone forwarders.

[4]In each of our experiments, each PlanetLab node becomes active at some random, independently chosen time during the first twenty minutes of the experiment.

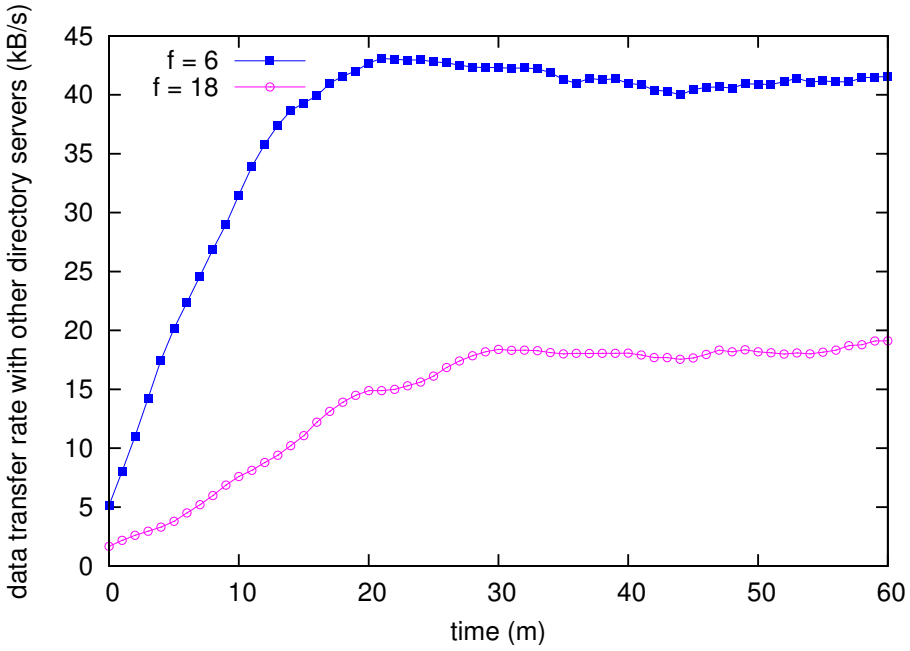


Figure 5.10: Inter-Directory Traffic Profile. *Five-minute moving average snapshots, by minute, for traffic from a typical directory server to its neighbors, given* $n_f = 6$ *and* $n_f = 18$*. We define* $T_d = 20$*,* $\delta = 8$*, and peering directive* `summarize`*.*

In Figure 5.12, we show the overall traffic volume of control messages sent between directory servers and standalone forwarders using peering rules `proxy` and `summarize`. Recall that the `summarize` peering directive instructs directory servers to **not** propagate forwarder records from a directory server neighbor but instead propagate lists of forwarders whose records are held by the directory servers indicated. The `proxy` peering directive instructs directory servers to aggregate all of the names of forwarders received from a directory server neighbor into a single list; i.e., a directory servers provide all of the forwarder names but not the directory servers that contain their records. When the `proxy` peering directive is used, clients are referred to a neighbor of the directory server if a satisfactory forwarder record is not found. Recall the inherent tradeoff between circuit performance and traffic volume to standalone forwarders, as

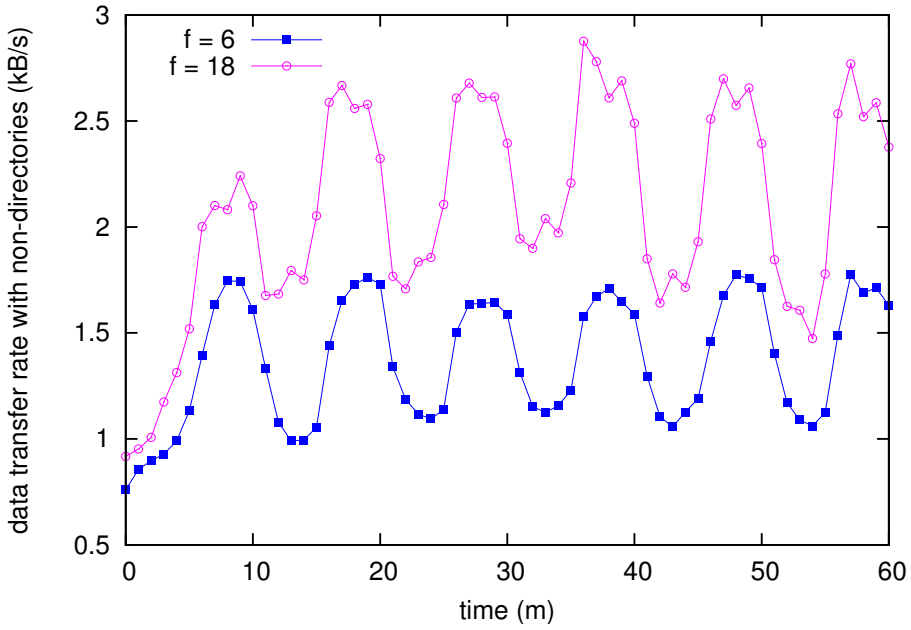


Figure 5.11: Traffic Profile between a Directory Server and Standalone Forwarders. *Five-minute moving average snapshots, by minute, for traffic from a typical directory server to the forwarders whose forwarder records are published directly, given* $n_f = 6$ *and* $n_f = 18$. *We define* $T_d = 20$, $\delta = 8$, *and peering directive* `summarize`.

described in Section 5.2.1. A network designer would consider this effect in selecting a peering directive.

Finally, Figure 5.13 presents a summary of how peering directives affect control plane activity. We conclude that peering directive `full` is probably too expensive to justify the decrease in circuit setup latency in large PANs, but we note that in small PANs, the `full` directive may be adequate. Peering directive `proxy` reduces control plane traffic quite substantially, but at a cost to circuit performance that may be prohibitive. Which peering directive to choose is inevitably a function of the constraints of the underlying network topology and the needs of client applications.

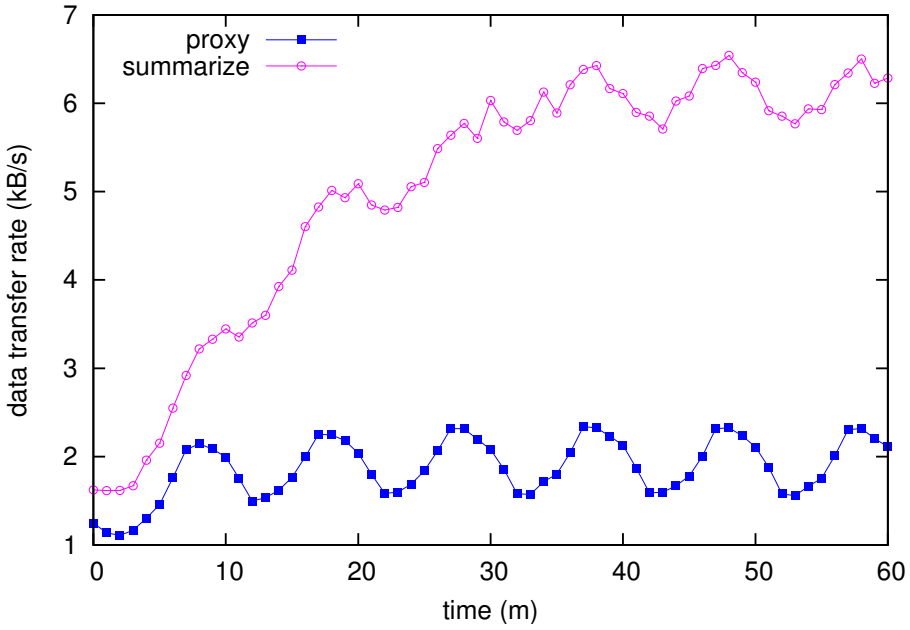


Figure 5.12: Traffic Profile: Proxy versus Summarize. *Five-minute moving average snapshots, by minute, for traffic from a typical directory server to the forwarders whose forwarder records are published directly, given* $n_f = 18$, $\delta = 4$, *and* $T_d = 20$, *using peering directives* `proxy` *and* `summarize`, *respectively.*

5.2.3 Comparison to Interdomain Routing

Two of the most important problems associated with BGP are protocol oscillations and security vulnerabilities (43). Both of these problems arise as a side-effect of the implementation of policy within BGP.

Protocol Oscillations

Routing oscillations occur as the result of conflicting preferences among autonomous systems. Indeed, the policy mechanisms of BGP allow the possibility of configurations that never converge. In particular, it is possible for a set of pairwise-neighboring autonomous systems, arranged to form a cycle, to have static policy preferences that

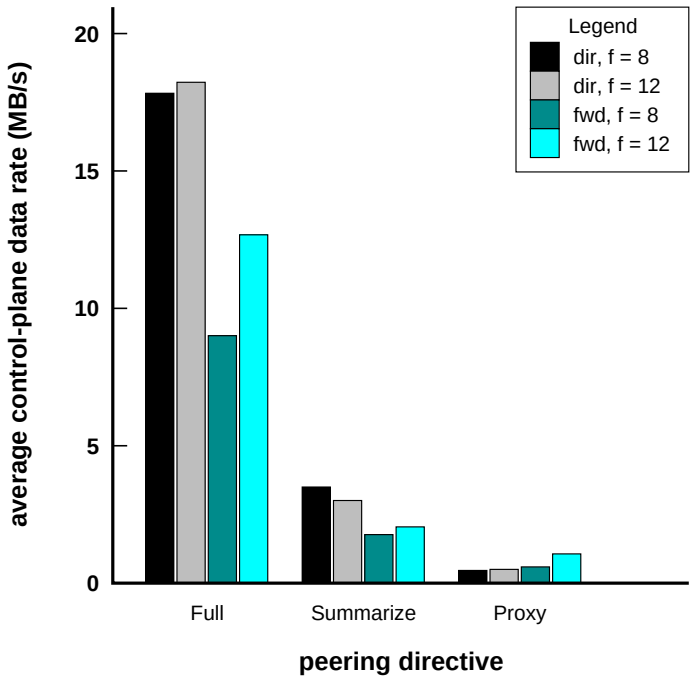


Figure 5.13: PEERING DIRECTIVE COMPARISON. *Effect of peering directive on traffic volume. We show examples for* $n_f = 8$ *and* $n_f = 12$*, given* $\delta = 4$ *and* $T_d = 60$*. Bars marked* `dir` *indicate traffic to neighboring directory servers; bars marked* `fwd` *indicate traffic to forwarders.*

cannot all be simultaneously satisfied. Griffin et al. refer to this configuration as a *dispute wheel*, and the result is an infinitely repeating sequence of BGP update messages (57). Whether such static policy configurations are fundamentally wrong or simply the result of an intrinsic dispute between the parties managing the autonomous systems is beyond the scope of our consideration. The significance of route oscillation is that the abundance of messages induces heavy processing load on individual BGP routers.

The policy language of PAN allows the expression of static policy configurations

that can result in similar oscillatory behavior. In particular, since our adapted version of RPSL allows a very general expression of preferences that depend upon the forwarding path, it is possible for dispute wheels to exist. We believe that the trade-off is worthwhile: preventing such configurations would require reducing the inherent richness of our policy language.

Another significant source of interdomain routing oscillation involves multi-exit-discriminators (MED) (58), to which PAN has no analogue.

Security Vulnerabilities

As BGP provides information for controlling the flow of packets between ASes, the protocol plays a critical role in Internet efficiency, reliability, and security. The Internet can be severely impacted by BGP failures. Accidental misconfigurations have resulted in serious routing problems and loss of service (82). However, failures are not always accidental—attacks intended to cause widespread outage on the Internet will (and do) target BGP (91; 123). Denial of service is not the only concern; an attacker might redirect the flow of some traffic through his network so that he can eavesdrop on it.

BGP has several well-known vulnerabilities. Neither the originating announcement of a route, nor the information attached to it as it traverses ASes are guaranteed to be correct. Moreover, BGP does not provide any way of identifying the source of bad data. Hence, misconfigured or malicious routers can, among other things, force other ASes to accept bad or inefficient routes, hijack address ranges, or simply flood the network with useless route information.

By requiring the PAN client to mediate the construction of circuits, we resolve some of these issues. Specifically, PAN clients have an assurance of the paths that traffic takes through the circuits that they construct. Also, traffic between the client and the last-hop forwarder is encrypted, so eavesdropping has limited use (except, perhaps, in compromising anonymity). However, the last-hop forwarder may still terminate individual connections, or even observe or modify unencrypted TCP streams. Since Tor does not allow circuits to be dynamically rebuilt after a TCP stream has been attached, Blossom in particular suffers the weakness that any forwarder in a circuit may fail, breaking the TCP connection.

5.3 Deployability and Incentives

Deployment of PAN forwarders offers numerous benefits; Section 5.4 describes these benefits in greater detail. However, a PAN cannot succeed with exit forwarders alone; the needs of individual organizations must also align with the incentives for deploying the network itself. Specifically, this means configuring and maintaining the directory servers that provide the core of the PAN infrastructure. The fear of legal liability associated with running PAN forwarders or directory servers may have a significant effect on incentives; we consider such challenges in Section 6.3. In this section, we consider the technical problems that a PAN might encounter as it scales, and we describe how the mechanisms in PAN can be used to address these problems.

5.3.1 Aggregation Strategies

Aggregation promotes scalability; one reason to not aggregate when possible is to reduce the time required for clients to find the perspectives they seek. Small PAN networks do not benefit from aggregation enough to offset the cost of increased setup latency, though as PAN networks expand in size, aggregation will become necessary to deal with the scaling issues. PAN provides the tools to perform aggregation where it is necessary for scaling, though for some semantic attribute categories, aggregation is not possible. In particular, hierarchically-organized categories (e.g., political location, network name, even geolocation) can by definition be aggregated. Flat categories, such as those describing filtering policy and functional capability, cannot.

Configuring directory server policy to aggregate hierarchical fields is straightforward. Refer to Figure 5.14. Observe that directory server DS3 receives perspectives located in various cities and then aggregates them all into a single announcement for `Canada.Quebec`. DS1 receives the aggregated perspective as well as additional perspectives from DS4. DS1 subsequently aggregates all perspectives from Canada into a single perspective.

In a PAN, individual perspectives may contain some number of attributes in each category (while political locations are mutually exclusive, filtering policies are not), and a user may ask for some particular combination of attributes. While we do not aggregate across fields to create the cross-product, we do allow individual directory servers to decide whether to *subdivide* a perspective that provides a particular combination of attributes, advertising the constituent attributes individually or in smaller sets. For example, a perspective that is located in Saudi Arabia and provides ac-

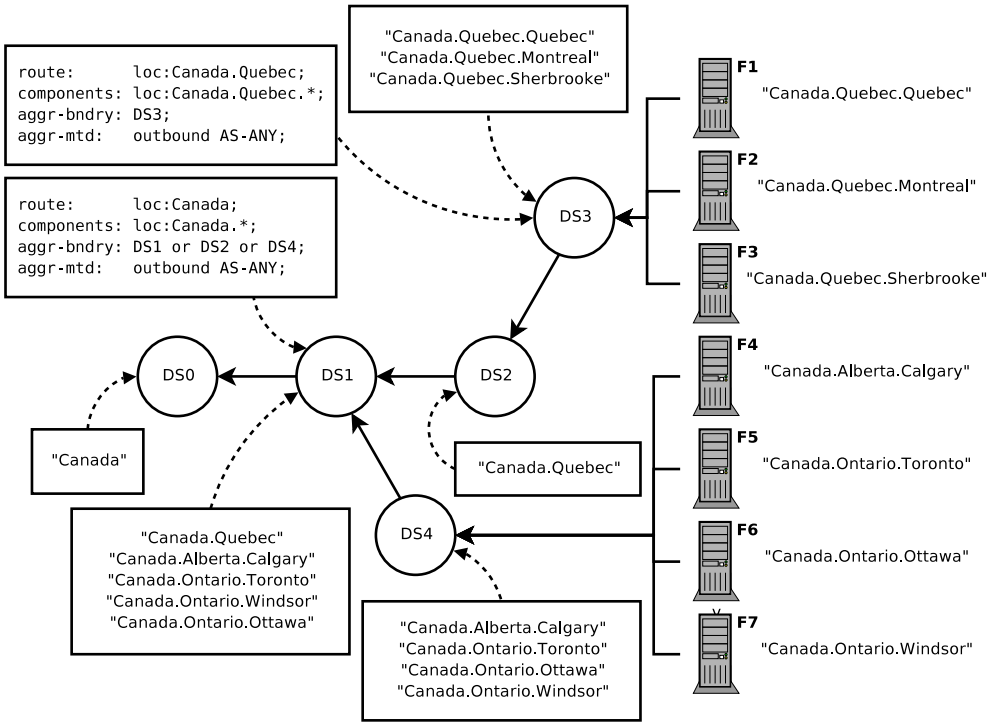


Figure 5.14: PERSPECTIVE AGGREGATION. *Certain metadata, such as political location and network name, are hierarchical and thus by definition aggregatable by directory servers. Newly created aggregate perspectives are assigned new, empty forwarding paths; the forwarding path associated with individual perspectives to be aggregated are ignored.*

cess to news stories might be advertised as two perspectives, one that is located in Saudi Arabia and one that provides access to news stories. Directory servers may use a *dynamic learning* procedure, as described in Section 5.5, to determine which combinations of attributes are most popular as a basis for determining which sets to subdivide.

Figures 5.15 and 5.16 present a scenario in which a series of forwarders advertise perspectives with various combinations of attributes denoting location in Iran and filtering policies that allow access to "Pro-Democracy," "Religion," and "News"

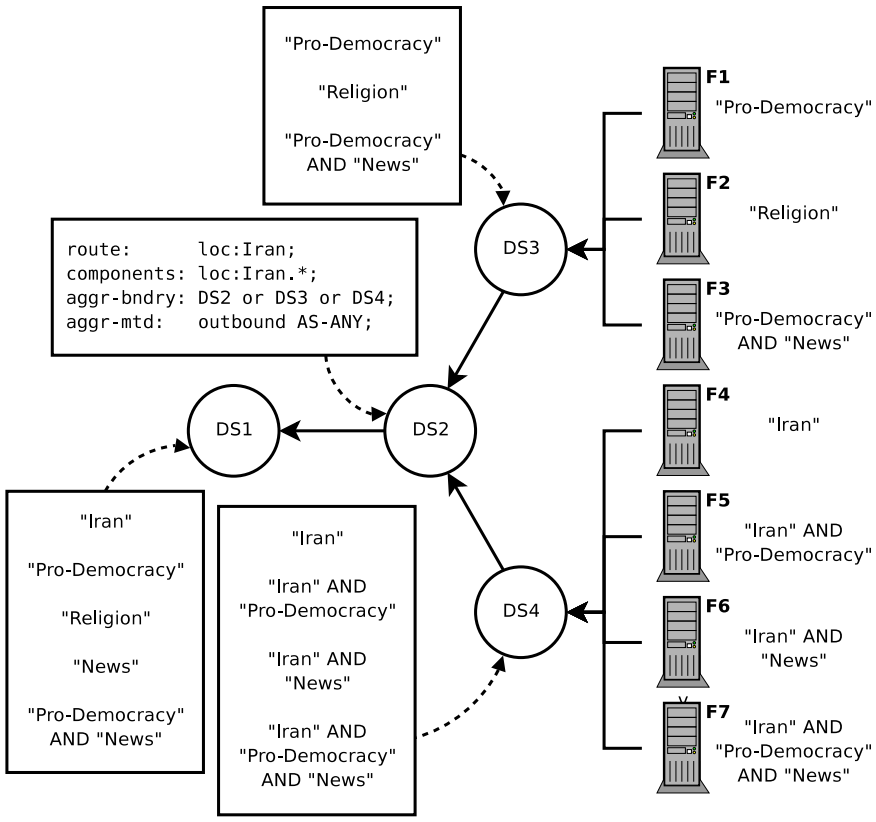


Figure 5.15: Subdivision of Perspectives (1). *If a directory server receives a preponderance of perspectives with different combinations of some set of attributes, it can reduce the number of perspectives that it advertises by advertising the attributes separately.*

content. In Figure 5.15, directory server DS2 advertises "Iran" separately but still allows combinations of the other attributes. In Figure 5.16, directory server DS2 uses a policy such that it advertises each attribute separately.

The tradeoff resulting from aggregation or subdivision is that clients are not guaranteed to get the perspective that they want in one querying pass through the network. See Figures 5.17 and 5.18 for an example. Figure 5.17 shows a client seeking a perspective located in Iran that provides access to "Pro-Democracy" content. While

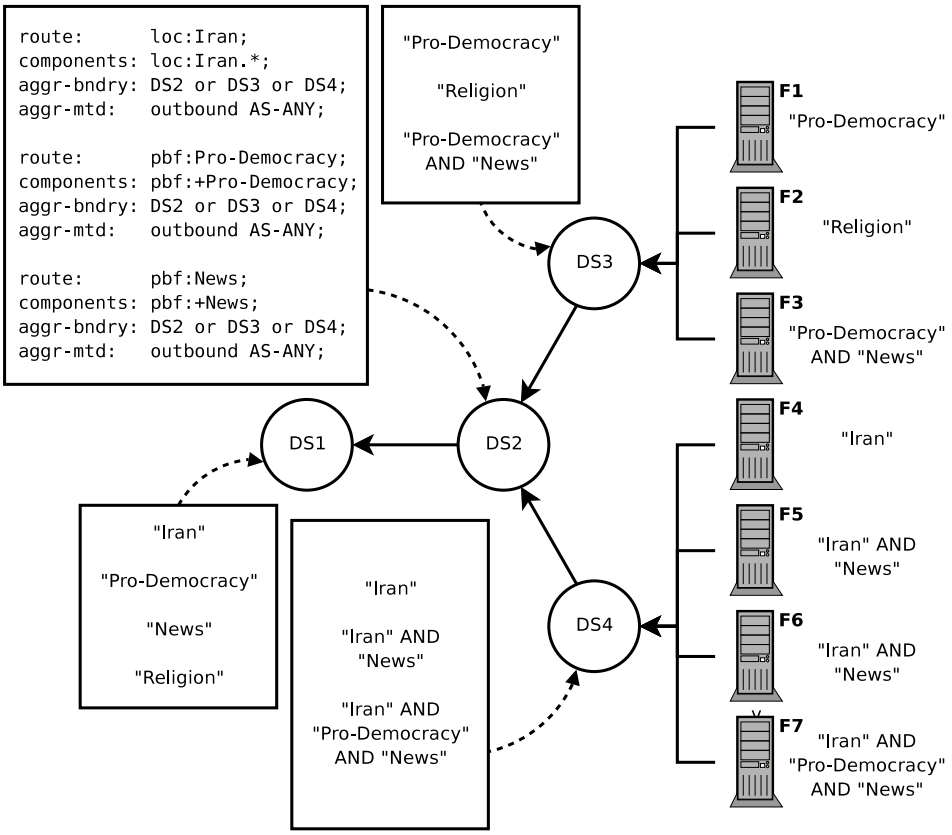


Figure 5.16: Subdivision of Perspectives (2). *Advertising attributes separately may dramatically reduce the number of perspectives to advertise. Note that DS2 has no aggregation policy for* `Religion`*; by default, directory servers do not perform aggregation.*

DS6a and DS6b both advertise that they provide both "Iran" and "Pro-Democracy" perspectives, only DS6b actually has knowledge of a perspective that provides both. When the client is in the process of learning the path, it is faced with a choice when it reaches DS3; suppose that it chooses DS4a as its next hop. Then, when the client reaches DS6a, it determines that the branch of the path following the decision point is invalid. The client then chooses the other path, and finds a perspective that matches its query (see Figure 5.18). We presume that after some number of unsuccessful

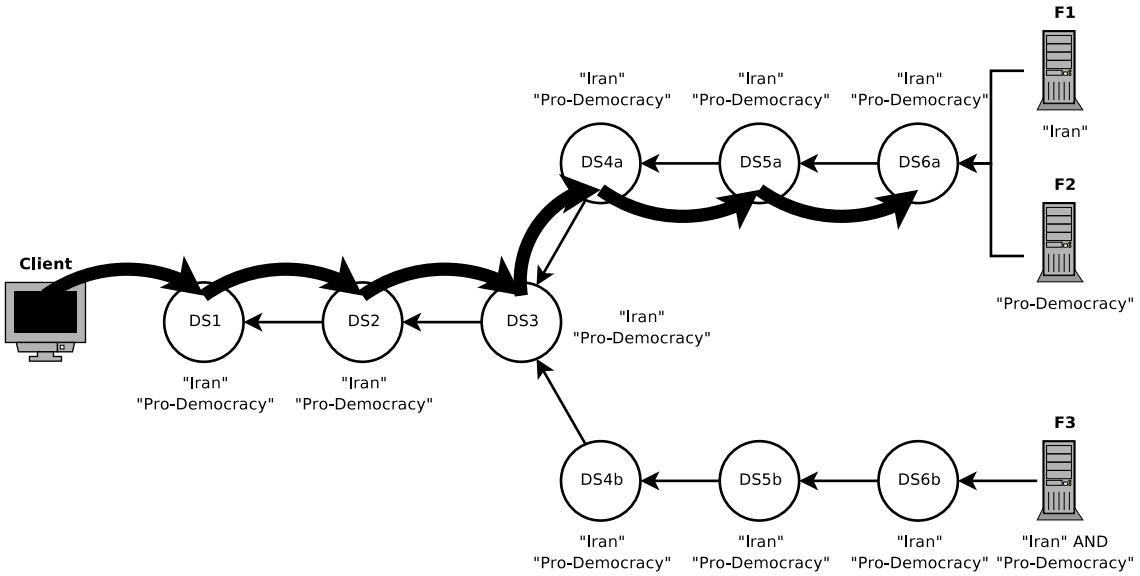


Figure 5.17: Choosing an Uncertain Path. *A client seeking a perspective containing a combination of attributes may issue queries along an incorrect path.*

attempts, the PAN client will abort and return an error condition to the application.

Observe that the client incurred a penalty for choosing the wrong path. Consider the following simple model that quantifies the penalty. Consider the directory server at which a client is faced with a choice among possible successive directory servers as the *decision point* (shown by DS3 in Figure 5.17), and consider the directory server at which a client learns with certainty the correctness (or incorrectness) of its circuit-building decision as the *aggregation point* (shown by DS6a in Figure 5.17). Suppose that there are n_d hops between the client and the decision point and $n_a(i)$ hops between the client and the aggregation point i. Let β denote the expected number of times that the client will have to backtrack before finding an acceptable circuit, and let n_a^* denote the average number of hops between the client and the aggregation point.

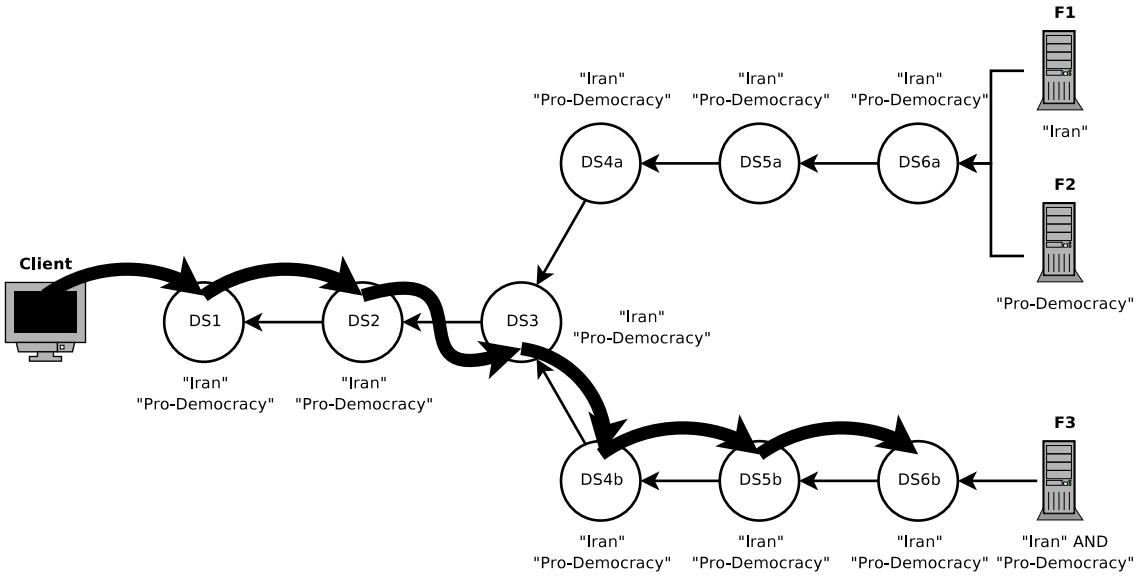


Figure 5.18: BACKTRACKING. *If, by querying, a client discovers that a chosen path does not lead to the desired perspective, it may backtrack to try a different path instead.*

Next, suppose that A and B represent attributes, and a client wants a perspective with both attributes, but attribute A is not provided in a single perspective with attribute B because of aggregation or subdivision. Suppose that the client finds a sequence of directory servers that advertise attribute A. Let $p(X)$ represents the probability of a given perspective having attribute X. Directory servers have knowledge of the number of entries with perspectives A and $B \cap A$, so:

$$\beta \approx \frac{1}{p(B|A)} = \frac{p(A)}{p(B \cap A)}. \tag{5.6}$$

Next, define $\tau(n)$ as the expected time required to build a circuit of length n. The value of $\tau(n)$ can be approximated by the quadratic regression curve given by Table 5.1 (and depicted in Figure 5.2). For simplicity, we assume that all aggregation points are at the same distance from the client. Note that the client need not backtrack all

the way to the start of the circuit, but only to the decision point, so each backtracking requires expected time $\left[\sum_{i=1}^{\beta} \tau(n_a(i))\right] - \tau(n_d)$. Therefore, the expected time t that a client can expect to spend constructing a circuit to a perspective containing both attributes A and B is given by:

$$t = (1-\beta)\tau(n_d) + \sum_{i=1}^{\beta} \tau(n_a(i)) \approx \tau(n_d) + \frac{p(A)(\tau(n_a^*) - \tau(n_d))}{p(B \cap A)}. \tag{5.7}$$

Whether aggregation is sufficiently desirable to outweigh the performance penalty is determined by the extent to which the impact on client performance outweighs the impact on directory service performance. In addition, it is possible for clients to improve upon the circuit setup time given in Equation 5.7 by considering multiple paths in parallel, but this improvement carries the potential for a substantial cost to directory servers and forwarders that must respond to unnecessary queries and build unnecessary circuits.

Finally, improvement over time in the technology of the forwarders themselves will continue to change the degree of aggregation that is required for scaling.

5.3.2 Resource Management Strategies

PAN infrastructures are quite manageable because the primary elements that need management are the policy filters configured on the PAN directory servers. The PAN policy framework is based upon a simplified subset of RPSL, a widely-deployed and well-understood language for describing the configuration of BGP routing. The framework augments RPSL to address the special requirements of PAN, which includes adding a more general set of metadata to describe perspectives and specify

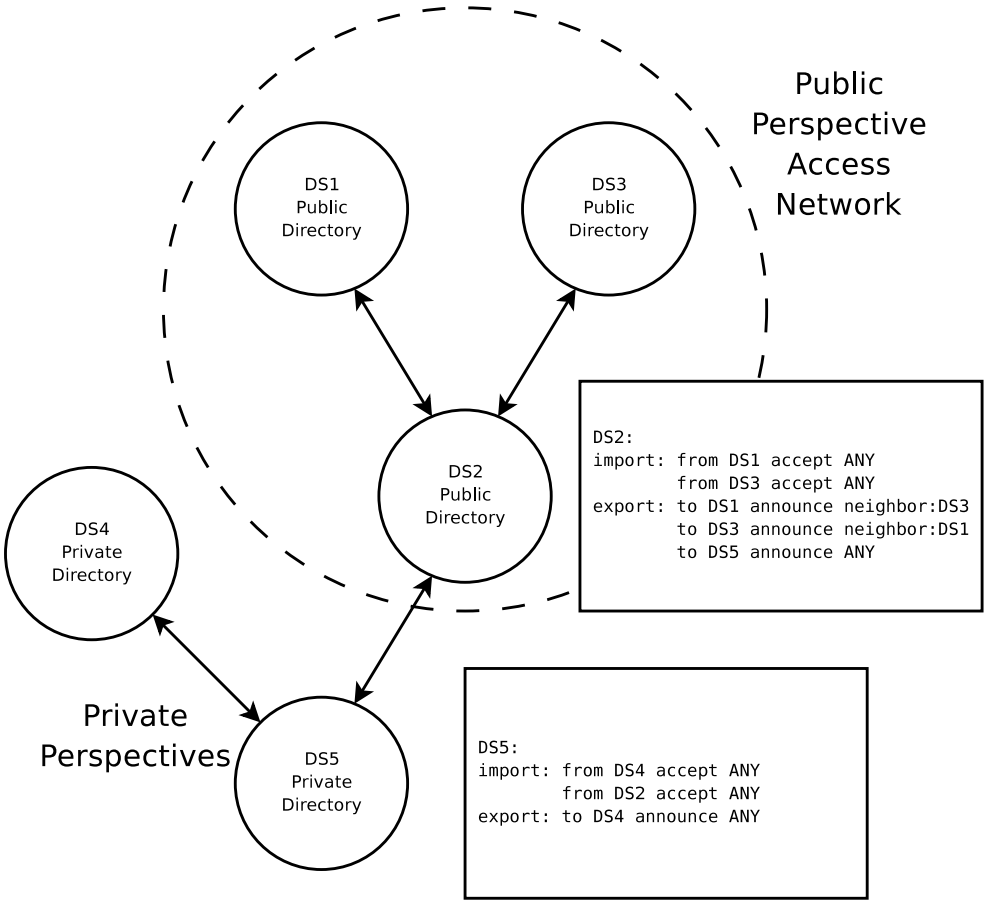


Figure 5.19: Filtering Policy. *An operator may want to configure a directory server to collect perspectives from two separate networks (for example, one public and one private) but only share information in one direction.*

when to subdivide attribute sets.

The policy language is used to configure the aggregation and forwarding policies of individual PAN directory servers, including a way of managing resource utilization as well (bandwidth limitations can be applied on a route-by-route basis).

Figure 5.19 shows an example of a filtering arrangement chosen to separate private directory servers from a public PAN. Suppose that directory server DS4 is part of a private PAN, but clients that consult DS5 require access to perspectives available

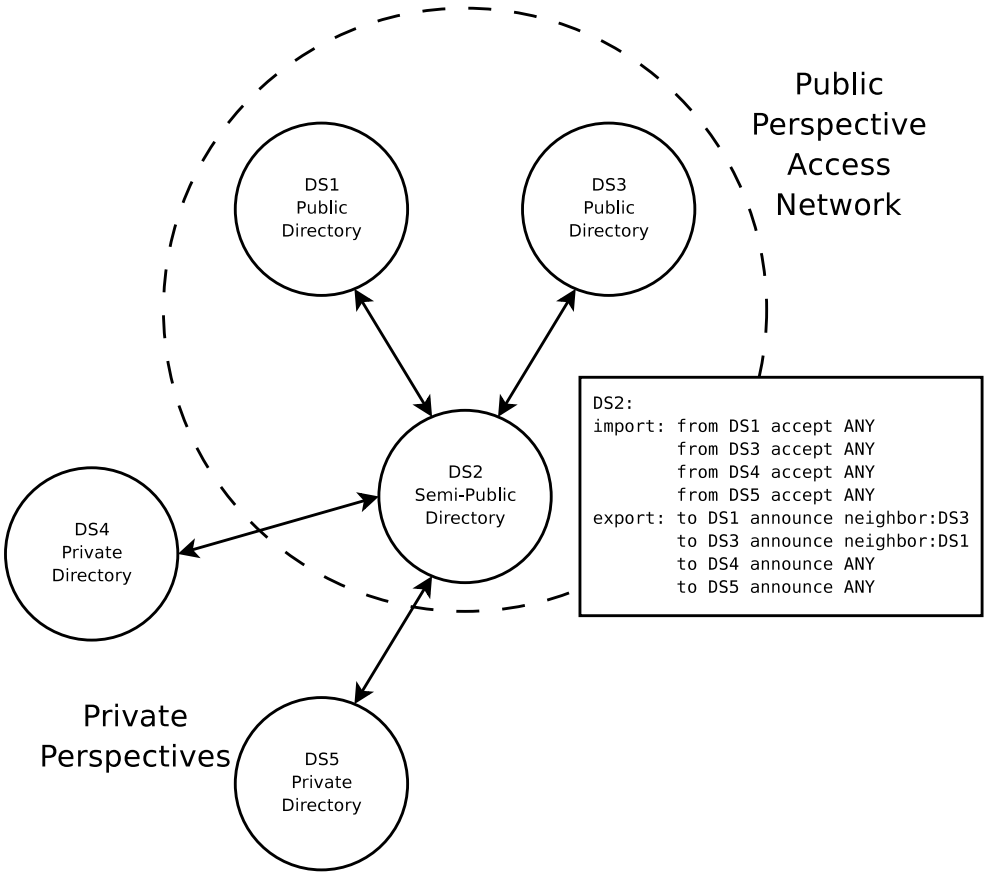


Figure 5.20: Semi-Public Directory Server. *An operator of a single directory server may want to participate in both a public PAN and a private PAN at the same time.*

via both DS4 and the public PAN (which is available from a combination of DS1, DS2, and DS3). Then, DS5 can establish peering relationships with both DS4 (for routes from the private PAN) and DS2 (for routes from the public PAN), with policies configured to not advertise routes from DS4 into the public PAN.

Another way to keep the private and public PANs separate is to operate DS2 as a *semi-public* directory, meaning that it can use circuit extension rules to assure that private routes remain private while still exchanging public routes. An interesting

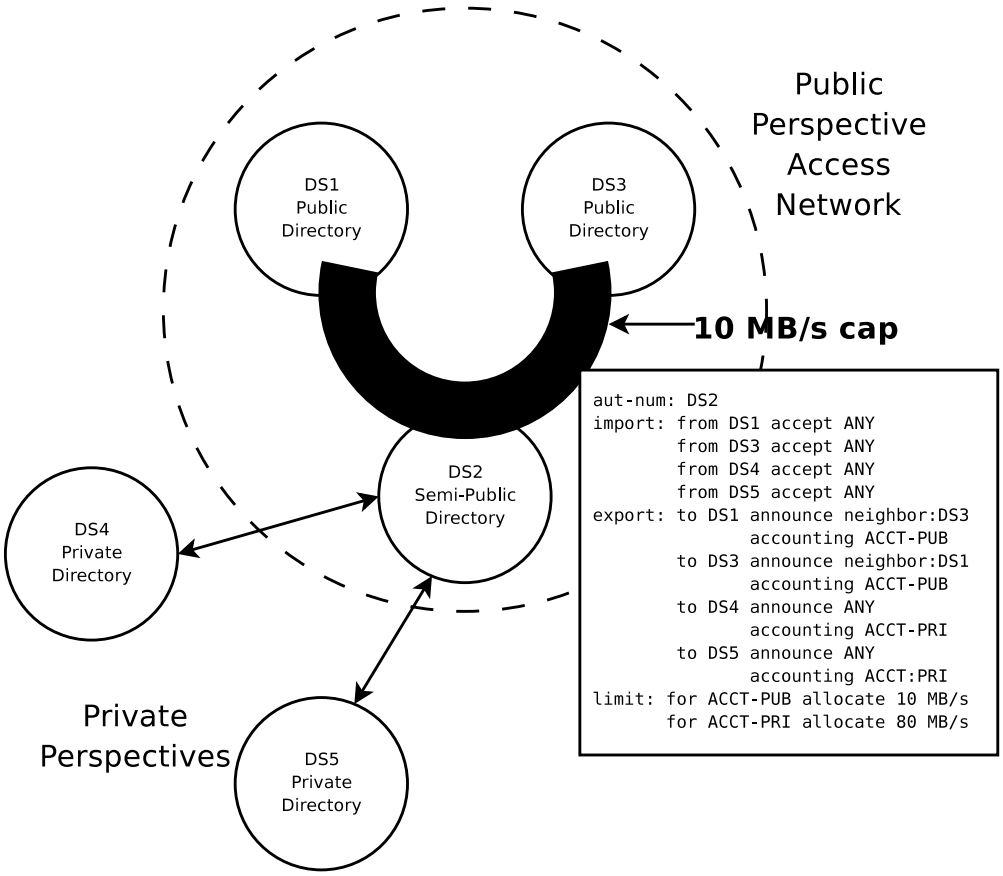


Figure 5.21: Resource Management. *Directory servers may be configured with accounting sets that impose bandwidth quotas on a per-route basis.*

feature of PAN is that a single directory server can serve both purposes. For example, in Figure 5.20, DS2 is explicitly configured to relay advertisements between DS4 and DS5 without sharing their routes with the rest of the PAN. Of course, this could mean subjecting DS4 and DS5 to bandwidth limitations associated with entrusting their conversations to a directory server that also forwards public traffic. Suppose that the administrator of DS2 wants to ensure that DS2 does not spend too much of its time relaying traffic between other nodes in the public PAN. Figure 5.21 shows how DS2 can establish resource management policies, as described in Section 4.4, to

set a bandwidth quota on traffic between DS1 and DS3.

5.4 Usefulness of Perspective Access Networks

It is difficult to judge the feasibility or usefulness of PAN via experiments or theoretical analysis. PAN cannot solve every problem. In particular, while scalability limits the use of PAN directory servers to describe perspectives in particularly fine-grained terms, the notion of plausibly universal access is an important guiding principle for PAN. Most likely, PANs containing perspectives that describe private networks will be privately managed, and the directory servers that advertise these perspectives may or may not have peering relationships with directory servers in a large, "general-use" PAN.

5.4.1 Essential Applications

To assess the most important applications of PAN, we focus on five essential uses of PAN. To bring the benefits of PAN into sharp relief, we consider the specific advantages that PAN offers over VPN-based solutions for each application.

- Circumvent Political Filtering. PAN provides a tool that can be used to promote human rights. Authoritarian regimes and network access providers sometimes monitor or restrict access to Internet content for political reasons. Parties interested in providing access to restricted content to dissidents and others can deploy PAN infrastructure so that people whose attachment points to the Internet ordinarily subject them to such monitoring or filtering can access

Internet content as if they were in other parts of the world. For example, in China, access to resources varies widely among ISPs, since there is no consistent policy that is applied centrally throughout China's backbone, but a set of guidelines instead (99). Thus, if an organization like Open Net Initiative[5] were to use Blossom to conduct clinical filtering tests, it would probably want both geolocation and jurisdictional location attributes. The primary advantage of PAN over VPN technology in this context is the directory, which enables users to generically describe the perspectives that they want without needing to know what particular hosts are providing the perspectives. The directory also offers some robustness benefit, since while individual servers that offer a particular perspective may join and part the network, the perspective itself may remain extant.

- ENTERPRISE. Organizations with multiple separate networks can use PAN to selectively extend the trust envelope to allow access across network boundaries. In particular, an enterprise may want to allow users to access an internal network segment in one branch office from another branch office. We provide an example in Section 3.6. The primary advantage of PAN over VPN in this context is the naming infrastructure: *resources* in one network fragment have a standard way of *describing* resources in other fragments. While in the analogous VPN setup a user would need to specify the appropriate VPN to use to access some particular content, PANs make all of these decisions implicit by linking all of the VPNs together into a single framework.

[5]Open Net Initiative, `http://www.opennetinitiative.net/`

- Geography-Based Personalization. Since we know that the Internet is not consistent, there may be a market for Internet perspectives. For example, website internationalization or targeted advertising are sometimes a function of geolocation. Travelers far from home may be willing to pay to view the Internet as if they were home, so that they can have some assurance that the content they find is relevant to their interests. Similarly, a user may want access to targeted advertising and customized searches available in a location to which that user is planning a trip. As with the political filtering case, the primary advantages of PAN over VPN in this context are the directory and the robustness benefits it provides.

- Distortion or Projection of Location. A user may have in interest in appearing to be somewhere else for the purpose of determining what is accessible from a remote perspective. This can be useful for performing security audits, as it provides a means of appearing to be on the other side of firewalls and other policy-enforcing boundaries. This use can also be humanitarian; for example, Open Net Initiative periodically publishes a series of reports cataloguing the extent and scope of Internet filtering in a number of nations (100). Such cataloguing requires perspectives from which to observe the filtering. As with the political filtering case, the primary advantages of PAN over VPN in this context are the directory and the robustness benefits it provides. If network transparency wanes in the future, then the routing capability will also become important, since upstream providers may eventually render certain network locations not directly reachable.

- Topology-Independent DMZ. An organization may want to externally provide some view of an internal part of its network, for example to provide access to some walled garden to the public or to industry collaborators. PAN provides the ability to provide an "internal DMZ" with all of the flexibility of remote access to a DMZ at the edge of the network but none of the topological constraints. A commercial application of this approach is gaming: individuals can use PAN so that they can share a local area network and play distributed games designed for a single LAN. The primary benefit of PAN over VPN in this context is the routing, since PANs offer a means of reaching *perspectives* that otherwise would not be directly accessible. Of course, it is possible for a large enterprise to construct a persistent tunnel from a VPN server to a publically-accessible network location in this case, but such an arrangement may not be desirable in all cases.

5.4.2 Security Considerations

Next, we consider the security implications of the PAN architecture. The circuit-based design sacrifices stateless forwarding in favor of path authentication and resistance against man-in-the-middle attacks. The salient advantage of PAN over VPN in resisting adversarial filtering is that a perspective can continue to exist even if an adversary filters access to some proportion of the PAN forwarders: in theory, as long as a path exists from the client to the desired perspective, PAN should be able to find a way to deliver the circuit.

However, the PAN infrastructure introduces some security vulnerabilities as well.

For example, providing additional infrastructure components within the network introduces new services that can be attacked. Adversaries may choose to operate rogue forwarders or compromise existing exit forwarders. With control of an exit forwarder, an attacker could potentially monitor or modify the traffic between the exit forwarder and the application server. Adversaries may also attack directory servers for the purpose of returning invalid or misleading query results, injecting bogus route announcements, or discerning and cataloguing which users are requesting which perspectives.

Another serious concern is that a determined adversary can systemically filter access to forwarders or directory servers within a PAN. This means that if a repressive regime decided to block access to PANs by determining the set of PAN forwarders, it could do so; there are important reasons for designing PANs such that the network locations of the forwarders and directory servers are public (refer to Section 3.1.3). Furthermore, if a repressive regime were sufficiently paranoid, it could block all encrypted or unapproved traffic, relegating the use of PANs to steganography or covert channels. While some projects aspire to provide covert channels, PAN itself does not. Fortunately, case studies have demonstrated that Internet filtering in China (in particular) is inconsistent (99), suggesting that China is either incapable or unwilling to systemically filter all access to circumvention technologies. For example, as of July 2006, the set of hosts not generally filtered by China includes most of the Tor network.

Considering the preponderance of incomplete attempts to filter access to Internet resources by category, we identify a set of useful countermeasures for dealing with a limited adversary. Consider an adversary that controls a network that traffic from

PAN users in a particular region of the Internet must traverse. One countermeasure is to reveal the network locations of PAN forwarders sparingly, perhaps configuring directory servers in some regions of the Internet to only provide a limited number of forwarder descriptors per unit time. The challenge is that providing public access to a circumvention system means providing access to adversaries as well, and if adversaries know how to reach parts of the network, then adversaries can block the network. Releasing network locations incompletely and slowly over time creates a race between adversaries and regular users of the system. The optimistic vision is that while the set of nodes providing gateway access to the system may change, the fact that users continue to have access will not.

A second countermeasure is to "multiplex" Perspective Access Network directory servers with servers that provide other, "innocuous" content that a network infrastructure provider cannot afford to deny to its users. Specifically, a popular website could offer access to a PAN as an indistinguishable part of its service, forcing adversaries to choose between denying their users access to this website and denying access to the PAN.

A third countermeasure is to use the latest techniques for establishing covert channels as a generic platform, and send PAN traffic over the covert channels. As mentioned in Section 2.6, Perspective Access Networks do not create covert channels, but this is not to say that they cannot interoperate with covert channels. Ultimately, PAN is not a complete solution for dealing with powerful adversaries seeking to deny access to circumvention technologies. However, it does provide a generic technique for describing which perspectives to access and constructing circuits to access these

perspectives; this technique may have greater value to users subject to the whims of powerful network-controlling adversaries once better covert communication techniques have evolved. In the meantime, we believe that the three countermeasures will provide significant benefits.

5.5 Scalability

For cases in which the number of directory servers and perspectives are less than a few thousand, the directory service can maintain the full set of perspectives. The enterprise, topology-independent DMZ, and geography-based personalization filtering scenarios described in Section 5.4 fall into this category.

However, an important scaling issue arises when the set of potential perspectives is large or unbounded. For example, describing perspectives in terms of the individual sites to which they have access is impractical since there are too many individual sites to maintain in a list. Similarly, providing a means of guaranteeing that a circuit can be built to some particular exit forwarder is also impossible for sufficiently large networks, since clients would only be able to refer to forwarders by their equivalence classes.

5.5.1 Case Study: Political Filtering

One possible application of Perspective Access Networks is circumvention of political filtering. This application is interesting because the number of possible attribute sets that clients can request may grow to be quite large in practice, and it is tractable because clients may tend to request only a small, relatively stable fraction of the re-

questable attribute sets. We suspect that most queries will include one policy-based filtering attribute combined with one location attribute, though certainly combinations of locations with multiple policy-based filtering attributes are possible.

Our challenge is to demonstrate that the sets of tuples (whether location-attribute pairs or n-tuples of location and multiple attributes) have enough stability to prevent excessive churn within the directory system. Excessive instability results when the bottom positions of the advertised ranking often fluctuate. There are two possible causes for such instability. The first cause is insufficient differentiation in popularity (i.e., shallowness of the popularity curve) among attribute sets. In this case, natural random sampling will cause rankings to vary widely, requiring large portions of the list to be replaced at each rebalancing. One way to reduce the impact of random sampling is to sample over a longer time interval (i.e., increase the rebalancing interval). The second cause of instability occurs when the set of popular attribute sets varies greatly over relatively short time-intervals, i.e., the set of interesting perspectives changes. One way to reduce the impact of change is to increase the amount of hysteresis.

We argue that the set of interesting perspectives is both sufficiently small and sufficiently stable that it is not necessary to increase hysteresis or the rebalancing interval to levels that render the system ineffective by keeping the list of popular attribute sets unnaturally static. First, we present an argument that the of kinds of resources to which a given jurisdiction has an interest in inhibiting access is likely to change fairly slowly with time. Second, we show that (a) the category sets implemented by popular filtering software are both small enough to be manageable and (b) the categories themselves are reasonably consistent across filtering platforms.

Indication of Stability from Existing Policy

According to Netanel, cyberanarchism asserts that users have the ultimate ability to choose the set sites that they want to visit (94). A cyberanarchist might argue, like Sieber (117), that filtering Internet content would be technically difficult if not impossible to do in real-time: data may be in arbitrary formats, data may be encrypted, etc. However, Netanel argues that users face substantial switching costs when moving from one "virtual forum" to another (94), and blacklists have proven to be effective in reducing access to content, even if they are not completely without false positives or false negatives. (Note that Perspective Access Networks potentially reduce the switching costs.)

In recent years, the Open Net Initiative has published a series of investigative reports characterizing the extent to which Internet content is filtered by each of about 20-30 repressive regimes worldwide. The Open Net Initiative devised a set of thirty controversial categories, and for each category, ONI found a few examples of popular websites and tested the reachability of those websites from Internet hosts in the various countries considered by the study. The results illustrated differences in filtering strategies among regimes that seek to filter Internet content. ONI has observed ISPs in Yemen to have substantial filtering rates for content involving sex, nudity, and drugs. Burma blocks sites providing email and pornography as well as important sources of news relevant to Burma. These categories do not seem particularly time-sensitive, though some forms of filtering are time-sensitive; for example, certain websites were inaccessible to subscribers of a certain ISP in Belarus on that country's 2006 election day (101).

Even within Western nations, there is substantial interest in generally restricting access to certain kinds of content available via the Internet. Ultimately, such interest generally involves a small number of categories. Indeed, we can enumerate these categories as surely as we can enumerate the various "nasty human impulses that are normally constrained by the sanction of collective morality" to which the Internet "holds up a mirror" (65). The official policy of essentially all Western countries includes restriction of pornography and opposition of child pornography in all of its forms. Restrictions in the EU follow the "slippery slope", transcending US restrictions by also opposing speech that incites "hatred, discrimination, or violence" (95), including regulations in Germany regarding speech about the Holocaust.

Even though the US Supreme Court has generally demonstrated an aversion to filtering Internet content on the grounds that it violates the First Amendment ("freedom of expression operates best in an unregulated marketplace" (95)), the US Congress has passed legislation such as the Communications Decency Act (1996), the Child Online Protection Act (1998) and the Children's Internet Protection Act (2000), all of which were intended to limit access to particular kinds of content available via the Internet and all of which impose the burden on Internet content providers. The content restricted by those laws is mostly limited to various forms of pornography and obscenity, and the stated intent is primarily to prevent minors from being able to access that content.

A 1997 law enacted by the government of Germany imposes similar restrictions on content, but goes even further: it declares that ISPs must either block these forms of content or provide customers with a device capable of the task, ostensibly because

the content is considered unsuitable for minors. The restrictions extend beyond those imposed by the US laws to include content that incites violence, contains hateful speech, or glorifies World War II (93).

In both the US and Germany, a study has shown that the general population ranks "pornography" and "protection of minors" as two of the four most important risks associated with the Internet, below "data protection" but above "fraud, manipulation." Most people surveyed also indicated that they categorically favored banning pornography and hate speech from the Internet (9).

The Platform for Internet Content Selection (PICS) presents an argument supporting the idea that filtering categories in the Western world can be expected to be fairly stable and long-lived. PICS aims to provide a standard language for describing content available via the Internet such that it can be reasonably filtered.[6]

Indication of Stability from Existing Mechanisms

Filtering technology, including filtering technologies deployed in several repressive regimes, generally consist of "commercial filtering products developed by U.S. corporations" (140). We argue that while the individual constituent hosts and URLs listed by these filters may fluctuate and change over time, the broad categories themselves are generally not time-sensitive. From our earlier discussion, it seems clear that the difference between filtering policies implemented by various regimes largely lies in the set of categories that they seek to filter.

Several years ago, CyberPatrol published a list of criteria for each of twelve content

[6]Platform for Internet Content Selection, `http://www.w3.org/PICS/`

categories.[7] WebSense also published a list of 31 categories as well as 50 subcategories to describe the rationale for listing the URLs that appear on its lists.[8] The category lists themselves are quite similar: a 2005 report contrasted four URL blacklisting databases (SurfControl, SmartFilter, Blue Coat, and WebWasher) and found 18 distinct categories that were directly comparable between the filters (17). Details varied only very slightly; the only significant differences among the filters were the particular URLs that they happened to filter, and, therefore, the relative effectiveness of the various databases as measured by the volume of entries in the filter.

While the sets of categories differ slightly among filtering systems, the categories mostly overlap, suggesting that the number of distinct categories is small. Also, the semantic descriptions of the categories do not seem time-dependent in any significant way. Combined with our (previous) assertion that a small number of new categories do not have a significant effect on the system, we assertion that the interests of governments remain largely constant over time, even if their degree of success does not. Considering our algorithm for inducing stability even in the event that some transient churn does occur, we believe that perspective churn will not prevent Perspective Access Networks from functioning effectively for the political filtering scenario.

5.5.2 Concrete Example

Research has demonstrated that the Internet filtering of China is the most sophisticated in the world (99). According to a 2002 Rand Corporation report, the Internet

[7]CyberPatrol Category Definitions: 1/20/99, `http://web.archive.org/web/20010405232448/http://www.cyberpatrol.com/cybernot/criteria.htm`

[8]WebSense URL Categories, `http://www.websense.com/global/en/ProductsServices/MasterDatabase/URLCategories.php`

filtering in China is is organized around three main content categories (22):

- Falun Dafa / Falun Gong
- The China Democracy Party
- Opposition to Chinese rule in Tibet and Taiwan

As an example, we consider the filtering of Web pages related to Falun Dafa by network access providers in China. Suppose that a person in China who wants to use a PAN to access Internet content related to Falun Dafa; this person has several options:

- 1. Ask for a location from which Falun Dafa would probably not be blocked (e.g., `USA`).
- 2. Ask for the relevant filtering-policy attribute (i.e., `+religion`) and assume that Falun Dafa would not be blocked from the resulting perspective, though it might be located in China, therefore exposing the person issuing the query to surveillance or prosecution.
- 3. Ask for a combination of location and filtering-policy attribute (e.g., `+religion` and `USA`), with an intent of viewing Falun Gong content from a society like the one specified.
- 4. Ask for a combination of location and a small set of filtering-policy attributes (e.g., `+religion` and `USA` and `+newsoutlets`) if the content happens to simultaneously fall under two categories that could be filtered.

Scenarios

For cases (1) and (2), we only need to worry about the network propagating single attributes, the most basic unit of perspective information. We assume that this information will propagate as far as policy permits. In general, when a client contacts its local directory server (DS) with a request for an attribute set, it receives a list of candidates for the next-hop DS (i.e., the neighboring DSes from which the DS being queried had received notification of availability of the perspective) in the circuit to be built toward a perspective matching the query. The client chooses one and iterates (without ever backtracking, as long as we assume that these perspectives have enough redundancy to not be subject to transient failures). Once the circuit has been built, the client attaches the application data stream to the circuit.

For cases (3) and (4), we argue that since queries for this combination of attributes are sufficiently popular, the combination itself is propagated unsubdivided through the network in the direction of clients that have given this combination of attributes its popularity. Now, when the client receives a list of candidates, it may receive next-hop candidate DSes that offer partial matches to the query. If it chooses one of these, then it must be prepared for backtracking. However, since we argue that perspectives are stable and the specific combination of attributes is sufficiently popular, the DS being queried should receive a complete match from its neighbors and thus backtracking will not be necessary once steady state is achieved.

Analysis

Finally, we use the considerations described in Section 4.5.1 to evaluate our Falun Dafa scenario:

- DRIFT. In China, filtering of particular sites and patterns often starts and ends, but the categories correspond to long-term political disputes and are stable.

- CHURN. The combined perspective (two or more (unlikely to be more than three) attributes total, as described in cases (3) and (4) above) will continue to be available as long as it remains popular (which it will, as long as China continues its filtering practice). Hysteresis will handle the case of China temporarily shutting off filtering for visiting dignitaries (as described in the RAND Corporation report).

- REQUESTS. As long as the chain of forwarders selected by the person in China enforces pairwise agreement that "religion" includes access to Falun Dafa content, then it will be sufficient to request the attribute `+religion` to access Falun Dafa content.

The same argument applies for the other two categories of filtering in China as well, though perhaps using some combination of `government` and `news-outlets` categories instead of `religion`.

5.6 Determining Attribute Categories

Our argument in Section 5.5.1 indicates that the number of attributes that are useful to describe actual Internet content filtering regimes is sufficiently small to be scalable. In this section, we address the concern about category specificity in PAN by presenting a method for determining what those attributes should be. We demonstrate that it is possible to have a list of categories that is both:

- sufficiently small to address the scalability (directory size, control plane messages) and management (ability for forwarders to conveniently describe their perspectives in terms of the set of attributes) issues, and
- sufficiently large to contain categories that afford clients the specificity they need to describe what they want to access (e.g., Falun Dafa is more specific than religion, but may still need to be described separately).

Consider the case of the a client seeking a perspective that provides access to Falun Dafa, as introduced in Section 5.5. The directory server presents the user with a list of attributes for which it can issue queries (and *attribute descriptions*, as described below). If one of the attributes is `Boston` and one is `+Falun_Dafa` (or an attribute whose description as provided to the client includes Falun Dafa), then barring the errors described below, the user will receive a perspective that provides access to Falun Dafa from Boston. If there is no *Falun_Dafa* category but there is a `Religion` category that does not include Falun Dafa in its description, then the user can proceed with the query but the kind of religious content available from the resulting perspective may or may not include Falun Dafa.

In the context of policy-based filtering, forwarders can advertise inaccurate perspective information for three reasons (we consider the example of a forwarder advertising "Religion" when it does not provide access to Falun Dafa content):

- Misunderstanding Attribute Definitions. A forwarder operator does not know that the "Religion" attribute actually requires the ability to access Falun Dafa content. To address this concern, we propose that neighboring directory servers periodically exchange "attribute descriptions", which are lists of content available via the Internet (e.g., URLs) that must be accessible for the perspective to be considered to have a specified attribute. A forwarder can download an up-to-date list of attribute descriptions from the directory server of its choice and determine what it can access from the attribute descriptions so that it knows how to describe its perspective in a manner consistent with what clients believe the various attributes to mean. Clients should query the directory service to learn the attribute descriptions during startup and periodically thereafter, so that they can present users with the list of possible attributes. Users then can determine the set of available attributes and build their queries accordingly.

- Errors. A forwarder operator knows what the "Religion" attribute means, but makes a mistake in describing the perspective as having that attribute. Unfortunately, systematically mitigating the impact of mistakes is impossible, though directory servers may "verify" forwarders before accepting them: Directory servers are presumed to trust the forwarders whose records they advertise. Just as certification authorities have a mechanism for ascertaining the validity

of the keys that they sign, directory servers may use the attribute descriptions to test forwarders before approving their attributes. (The mechanism for testing may, for example, involve verifying SSL certificates, or verifying a hash of the content of a snapshot of a web page as provided by a third party.)

- Purposeful Inaccuracy. A forwarder operator knows what the "Religion" attribute means, but deliberately decides to advertise it anyway, despite knowledge that the forwarder does not have access to Falun Dafa content. There is no clear way to distinguish this from the "Errors" case.

Depending upon the nature of the inaccuracy, directory servers have two means of resolving purposeful inaccuracy:

- Directory Server Pushback. If clients report (or directory servers observe through testing) a preponderance of forwarders that claim an attribute but fail to provide access to resources with that attribute, the directory servers may respond by not accepting advertisements about that attribute from the neighbor through which those forwarders are accessible.

- Creation of New Attributes. Suppose that the inaccurate forwarders are systemically failing to provide access to some well-defined subcategory of resources (like resources related to Falun Dafa under the `Religion` category). Then, the directory server can advertise (and describe) this new category (e.g., `Falun Dafa`) in the list of attribute descriptions that it periodically sends to its neighbors (neighbors are not obliged to accept it). Directory servers may also create new categories in this fashion, and they may drop categories (for scala-

bility) by simply not advertising them in the list of attribute descriptions. The process of creating and destroying attributes can be human-mediated (creating new categories in response to complaints from clients) or semi-automatic (in which the system checks for systemic access failures).

At a high level, whether `Falun_Dafa` ought to be a category of its own, separate from `Religion`, is not obvious. If the Internet consisted of only two distinct kinds of perspectives (perhaps one that filters content and one that does not), then we would need only one attribute to describe the perspective. The reason that greater richness among the categories is needed is that there are regions of the network that provide access to some categories of content but not others, and we require a means of describing these differences. For example, in the case of religious content filtering, we know that there are Internet regimes that block content related to Islam and **not** content related to Falun Dafa, as well as Internet regimes that block Falun Dafa and **not** Islam. By specifying a broad category such as `Religion`, we lose the benefit of all of the perspectives in the disjunction of those two categories. Conceivably, if the proportion of forwarders in locations that provide access to one but not both is sufficiently large, then the clients would benefit from having distinct categories. Otherwise, having two separate categories is detrimental as a result of increased table size and overhead.

Chapter 6

Conclusion

From commercial uses to human rights, Perspective Access Networks have a variety of interesting applications. They also provide an argument in the ongoing tussle between supporters of network neutrality and those interested in regulating access to Internet resources.

This chapter summarizes our work, assesses its social context and significance, and provides speculation about the future role of Perspective Access Networks. The first section summarizes our specific contributions. The second section characterizes the clear and present threats to network neutrality posed by the use of network location to identify and classify Internet users. The third section presents our vision of the legal and economic ramifications of the deployment of a general-purpose system that allows Internet users and service providers to share perspectives. The fourth section introduces some opportunities for future research projects in the space. The final section contains closing remarks.

6.1 Principal Contributions

If nothing else, PAN presents a new and useful way of considering the organization of Internet services and Internet connectivity. Our work provides the following contributions to the design of large-scale, general purpose internetworks:

- 1. Definition of Problem Space. The argument for PAN derives directly from real-world concerns. We introduce the *technical* issues, considering Internet history (79), design principles (29; 18), and conflicts of interest that naturally arise as the result of technical choices (30; 28). We also outline a variety of *social* issues: some socially conservative governments have required or implemented filtering within the network to restrict access to politically sensitive content. Additionally, some content providers use geographic origin to restrict access to commercial content by country; both commercial[1] and educational (110) content providers have been known to do this. While some of the more serious scenarios, such as systemic blocking of content by network access providers for commercial reasons and large-scale differences in access to content within and among Western nations, are not entirely realized, these and other concerns are quite real and have recently received substantial media attention.

- 2. Exploration of Solution Space. To position our work in the context of existing research, we provide an exploration of the most significant work in fields related to routing around network obstructions. We show how the key insight of our perspective-oriented approach differentiates Perspective Access Networks from prevailing approaches to addressing well-established problems.

[1] ABC, Inc. streaming service, `http://dynamic.abc.go.com/streamin`

- 3. Network Architecture. We specify a set of design goals and desiderata for a system that provides access to resources from different perspectives, and we consider the relevant tradeoffs associated with our specific design choices. We enumerate the requirements for a substrate that provides circuit-building and data transport functionality to our service. We suggest a number of potential applications of our technology, and we show how these applications make use of the specific features that we choose.

- 4. Directory Service. The PAN architecture lets clients consult a directory service to receive instructions to construct source routes for their application data. With scalability and ease of deployment in mind, we propose a directory service for our infrastructure. We provide a detailed specification of the functionality and behavior of this service, including both communication within the control plane as well as interaction between directory servers and clients.

- 5. Policy Framework. PAN extends the well-known routing policy configuration language, RPSL, to provide a policy framework that allows administrators of PAN directory servers to manage filtering and aggregation. RPSL is designed for use with BGP (4), and we consider how the policy needs of BGP differ from the policy needs of PAN. We specify revisions to RPSL based upon these differences, and we provide a number of examples to illustrate how our policy language can be used to meet the needs of administrators of directory services.

- 6. Implementation. We introduce Blossom, an implementation of PAN

suitable for actual use. Blossom relies upon *Tor* (38), a source-routing overlay network for TCP streams, for its circuit-building and data transport. While the goal of the deployed Tor network is anonymity, and our primary objective is access to resources from client-specified perspectives, we present an argument in favor of the appropriateness of the Tor software and its controller interface in meeting the PAN transport-layer requirements.

- 7. Empirical Results. We provide some empirical measurements of Blossom to assess the feasibility and dynamic behavior of PAN. The client performance measurements include measurements of throughput and latency of forwarders within the Tor network as well as a detailed evaluation of circuit setup performance using the PAN directory service. We also provide directory service measurements, evaluating the system in terms of size of forwarding tables, size of control plane messages, and frequency of control plane messages. In addition, we perform some experiments to illustrate the dynamic behavior of the control plane.

- 8. Strategies for Deployment and Scaling. Techniques for performing filtering and aggregation become necessary to ensure scalability of PAN. We speculate about how providers of PAN directory servers might want to use filtering and aggregation to their advantage, and we show how our policy framework provides the expressiveness to allow configuration consistent with their objectives. We also provide a mechanism for resource management, so that directory servers can provision bandwidth on a per-route basis.

6.2 Misuse of Location Information

Perspective Access Networks present additional challenges to service administrators who seek to use network location in fraud detection and abuse prevention. In particular, the use of network location to draw conclusions about users has become quite commonplace on today's Internet. Numerous institutional subscription services use IP address as the exclusive means of identifying users. IP addresses are sometimes also used as a criterion in fraud detection and abuse prevention, for example, flagging discrepancies between geographic location associated with an IP address and that associated with mailing address for credit card billing. For abuse prevention, many websites that allow public contributions (e.g., wikis, blogs, chat rooms, etc.) simply block the IP addresses from which chronic abuse emerges. Using IP addresses to categorize or identify users seems like a reasonable approach in general, but there are some important caveats as well. We briefly examine the long-term architectural dangers as well as short-term policy risks as we strive to put the costs and benefits of using location information into perspective.

6.2.1 Practical Justification

There is little doubt that some ISPs are more vigilant than others in curtailing spam and abuse. Many network administrators have accepted the idea that it is easier or more effective to fight spam by fighting the act of sending spam instead of addressing the security vulnerabilities that make spam feasible or the market forces that make spam desirable. At least for the present, knowing the ISP from which a connection originates certainly provides some statistically meaningful information

about whether the user responsible for the connection is likely to engage in antisocial behavior. Similarly, teaching Bayesian spam filters about network location may enhance their effectiveness in reducing spam.

Providers of online subscription services can be reasonably assured that most connections from IP addresses assigned to institutions that restrict access to their systems are from authorized users of those systems. Also, the overhead of setting up a mechanism that uses this information is far simpler than most alternatives.

The extent to which IP addresses can be used to discern geographic location is limited. For example, AOL uses large-scale proxy networks, and the IP addresses from which traffic from AOL subscribers originate (sometimes) do not carry fine-grained location information.[2] In addition, network providers that use 3G, the communications standard used for packet-switched cellular telephone networks, generally do not assign IP addresses geographically; as a result, emergency services for such networks use global positioning system (GPS) receivers or timing analysis to determine location (115).

Nonetheless, in recent years, a market has emerged for so-called "geolocation" services, which provide a mapping from network location to physical, geographic location. Service providers collect data from Internet service providers and resell it to geolocation customers in the form of datasets or permission to execute queries on their databases. Geolocation products have been developed for several uses, including fraud resolution, spam mitigation, targeted advertising, and digital rights management.

An important premise of geolocation is that there is much to learn about individual Internet users from how they are connected to the network, and such information can

[2]AOL Proxy Info, `http://webmaster.info.aol.com/proxyinfo.html`

be used as a basis for implicitly categorizing users by risk level or market segment. Geolocation affords advertisers the possibility of offering products and services specific to particular localities. Content providers wishing to disallow people from certain countries or regions from accessing certain content may, to a significant extent though not completely, use network location information to achieve this goal.

Similarly, it is reasonable that credit charges from locations that are far from the home of a credit card holder and that are known to be hotbeds of credit fraud merit close scrutiny. But this can have problems too: a large percentage of credit card use is associated with travel. Also, the preponderance of mail-order catalogs and (still easier to distribute) web pages allow even small local shops with only telephonic credit transaction clearing to have many remote customers. The credit industry typically errs on the side of giving users easy access to its services rather than denying undesired access when these goals conflict. The content industry typically adopts the opposite approach. For this reason, the IP address from which traffic originates is only one of many factors in authorization and identification for credit approval and fraud detection rather than an absolute or sole discriminator.

So far, it seems that the primary incentive for those who use network-layer information for application-layer decisions is to provide an expedient means of authorization that strikes an acceptable balance between easy access for desired usage and adequate deterrence against undesired usage. In short, it works. To date, IP addresses have been a resource difficult enough to obtain or spoof that they have fulfilled this role in authorization and fraud detection. More proper authentication, however, would require user certificates, a PKI, or other mechanisms that have proved

difficult to set up in large, relatively open contexts and have not seen widespread user adoption where they do exist. And, as long as end user systems not under institutional control remain as vulnerable as they now are to root-level intrusion, end-to-end authentication could also be an illusory approach to security. However, that is a much broader and separate problem than the one we are considering.

6.2.2 Immediate Side Effects

The ability to block abuser IP addresses is a powerful but ill-suited tool for some of the problems to which it has been applied: in a few cases, individual sites have blocked access from IP addresses in a broad geographic area. Two well-known examples are the blocking by a major-party presidential campaign of its web site from IP addresses outside the US just prior to the 2004 US presidential election and the blocking of 2004 Olympics coverage from IP addresses inside the US. In 2002, Pennsylvania ISPs blocked access to 1.6 million innocuous Web sites in an attempt to satisfy a state mandate intended to curtail child pornography (12). More recently, the major US ISP Verizon was the subject of lawsuits when it began blocking all email from Europe and other continents by default as a spam deterrent (81); since that time, Verizon has agreed to a settlement.

In general, infallibly binding the identity of users to how they happen to be connected to the Internet is not only impossible, but also undesirable. Proxies, workarounds, dynamic addresses, mobility, system vulnerabilities, and other complications make network location useful as a heuristic at best. To the extent that it is useful, the ability to use network location as an indicator of identity is a technical

shortcoming of the current Internet that can be overcome.

One of the main drawbacks to using network location for authorization is that legitimate users cannot access a service when away from their home institutions. It is possible to set up tunneling such that their accesses appear to originate from an IP address within the permitted range, but this may be onerous, technically difficult, or simply not possible as a matter of policy. A major expense associated with deploying and maintaining virtual private network (VPN) infrastructures derives from the need for individuals and businesses to access Internet resources that rely upon network location information to differentiate between valid and invalid users.

Furthermore, abusers can use proxies to connect through unblocked locations fairly easily. Arguably, Perspective Access Networks accelerate this process, but it is already happening and ultimately unavoidable. Individual proxies themselves can be blocked, but with multitudes of newly compromised hosts emerging daily, the effectiveness of that approach is limited. Networks designed to protect honest users from traffic analysis such as Tor (38) can be blocked because they explicitly provide a means of doing so, but abusers can take advantage of million-node botnets with no easily discernible pattern of IP address source. The result of network-based authorization, then, would seem to block the honest user from protecting herself while leaving the abusers unblocked and harder to find.

6.2.3 Long-Term Security Risks

We have noted some immediate practical problems, but solutions that avoid these problems raise additional security concerns of their own. IP tunneling simply to allow

use of institutional subscriptions when it is not otherwise needed is an extreme solution that may open the possibility of other intrusions to the institution. University librarians, among others, have long recognized the problem that authorization by IP address poses for remote users trying to access institutional services. This has been one of the prime motivators for development and increasing adoption of systems such as Shibboleth[3], which provides single sign-on and user-controlled credential management independent of IP addresses.

If legitimate users of credit systems have incentive via easier authorization of their transactions to route their traffic through an IP address associated with their home location, then they reveal via routing information their interactions with merchants and financial institutions not only to those principals but to observers as well. To the extent that users depend upon firewalls as a substitute for vigilance, installation of firewalls may leave them more vulnerable to identity theft, spear phishing, and the like. And, as actual large-scale systematic fraudsters become aware of the use of authentication by IP address, they are provided with specific incentive to spoof authorized or trusted (i.e., low-fraud) locations, or worse, to break into systems in or near those locations. This approach thus has the potential for greater vulnerability and risk for the legitimate user with a false sense of protection against the actual adversaries.

[3]Shibboleth, `http://shibboleth.Internet2.edu/`

6.2.4 The Role of Network Access Providers

The fact that network location provides an effective way to assign blame for malicious activity raises questions about the extent to which network access providers ought to be responsible for the comportment of the systems for which their networks serve as attachment points to the Internet.

Regulators have substantial interest in supporting the principle of enforcement within the network, since the network could potentially provide convenient points of control for execution of policy. Furthermore, lobbying by major telecommunications carriers generally encourages a movement away from uniform, open access and toward vertically-integrated "silos" in which carriers determine the set of resources that customers may access (14). So, there exist strong industry and regulatory forces to empower network operators at the expense of network neutrality and end-to-end connectivity. A recent ITU-T proposal advocates expanding the role of ISPs to require that they ensure that traffic traversing their wires adheres to certain normative requirements (102).

6.2.5 Function Creep and Expedience

Making use of the routing infrastructure itself to protect participants from each other is an arms race that sacrifices as collateral damage the neutrality characteristics of the Internet that provide its principal advantages over alternative interconnection paradigms.

The consistent response that those proposing such methods offer to proponents of end-to-end services is that it is too late to salvage the end-to-end principle and

that compromise is in order. Indeed, network-layer techniques have shown promise as expedient short-term remedies to exigent security threats, and as a result, governments and regulators have called upon ISPs to implement technical solutions within the network (149). Vint Cerf recently argued that, as far as security is concerned, it does not make sense to use the network to compensate for operating systems that protect themselves inadequately.

> According to Cerf, the more you ask the network to examine data—to authenticate a person's identity, say, or search for viruses—the less efficiently it will move data around. "It's really hard to have a network-level thing do this stuff, which means you have to assemble the packets into something bigger and thus violate all the protocols," Cerf says. "That takes a heck of a lot of resources." (129)

In other words, if we start requiring the network to perform tasks other than routing, then we undermine the ability of the network to do its most essential job. Another problem with such function creep is that it can become entrenched. Once a technique such as authentication by IP address is widely established, if legitimate technical reasons to substantially change how addressing and routing is done should arise, then they may be harder to implement and establish because existing systems are hamstrung by the use of IP addresses as authenticators or anti-fraud mechanisms, even if that usage was originally introduced as only an expedient.

6.2.6 Separating Identification from Routing

IP addresses were introduced to allow routing of IP packets. As we have already seen, if they are used for other purposes, and if identification and authorization become conflated with routing, then the purpose for which they were designed is un-

dermined: both legitimate users and attackers end up using IP addresses not because of routing, but to appear as authorized users. Onion routing was introduced ten years ago as an infrastructure that "separates identification from routing" (108). "Parties are free to (and usually should) identify themselves within a message. But the use of a public network should not automatically give away the identities and location of the communicating parties" (53). Anonymity from one's communication partner is not the primary motivation for Onion Routing; users may simply need to protect their points of attachment from attackers, whether personal (e.g., stalkers or identity thieves) or enterprise (e.g., corporate competitors gathering intelligence). In each case, the security benefits of separating routing from identification are substantial, even if the challenges it poses to the security models of some services are similarly great.

How will increased user mobility, increased use of anonymization networks for security by honest users, etc. interplay with the use of IP-address information for authentication and authorization? Intuitively, it seems that these two technologies are headed for a clash.

6.2.7 Discussion

The use of network location information in authentication, abuse detection, and fraud mitigation will have a substantial impact on the Internet environment for the next several years. Adversaries may or may not adapt to these techniques before the techniques become entrenched in the architecture of critical services. However, if history is a guide, they will adapt at some point, and more quickly if IP-address

location technology increases their incentives to do so. If we are to avoid arriving, therefore, at an entrenched burden with no ultimate benefit, we must understand the technology that we are using to do the job. In this section, we characterized some of the unanticipated ways in which relevant technologies for using network location interact. One response to understanding this is through governance and policy, but our focus herein is the technology itself. The complexity, brittleness, and overhead involved in the deployment of solutions that use network-layer address for authorization may stifle innovation in the future, even if each individual step along this path seems reasonable. But the news is not all bad: systems like Shibboleth and Perspective Access Networks provide a technological path that can continue to lead institutions away from authentication by IP address. Similarly, since network location is only one of many factors considered by fraud detection systems used in the credit industry, the technical framework already exists to allow an abandonment of network location as a factor as it diminishes in significance. This need not mean the end of geolocation services either; for example, people will still want to know about nearby restaurants and services. Geolocation services will just need to be based on information other than IP address if they are to continue serving a useful purpose. If the already existing security and functionality problems arising from IP-address-based abuse deterrence do not lead to its abandonment, then the incentives it provides to network attackers ultimately should.

6.3 Legal and Economic Effects

Large-scale, public deployment of Perspective Access Networks could potentially have significant legal and economic effects. For example, the uses we described in Section 5.4 are quite beneficial. Additionally, PAN may have value in promoting end-to-end security models within both enterprises and the Internet at large. As described in Section 6.2, some enterprises and providers of Internet services use network location for security purposes, either as an authenticator or as a basis for an assumption that the traffic is not exposed to the public. The authentication system we propose in Section 3.5 may be useful as part of a migration path from location-dependent to end-to-end security measures.

However, there are also risks, commercial factors, and chilling effects that could potentially cause influential parties to discourage large-scale deployment and use of PAN. For example, many service providers actually intend to use network location as a means of differentiating and categorizing users, and deployment of Perspective Access Networks has the potential to confound their efforts. Of course, open proxies can be used to circumvent geography-based access restrictions today, but the proxies themselves are generally considered illegitimate because they usually run on compromised or misconfigured hosts. PAN could potentially bring circumvention into the mainstream, and once this happens there could be calls for ISPs to implement policies that disallow the operation of PAN forwarders.

Perhaps the most serious threat to network neutrality involves the possibility that ISPs might filter or restrict access to Internet content for commercial reasons. Indeed, Edward Whitacre, the CEO of SBC, has even suggested the possibility that

both providers of content (e.g., Disney) and providers of services (e.g., Skype) ought to compensate the ISPs of their target audiences (98; 13) as part of a business model reminiscent of the cable television industry in the US. Clearly, the idea that ISPs should have the power to arbitrate which subset of the Internet to provide to its customers is very much alive. In fact, research has indicated that it is in the best interests of network providers to use compensation from content providers as a basis for discrimination among content providers, providing customers with inferior access or even no access to sites hosting particular content (137). While network neutrality regulations have certain costs, there is little else to prevent ISPs from selectively discriminating.

In the context of the Internet governance argument described at the beginning of Chapter 1, Clark et al. suggest that a tool that allows Internet users to circumvent provider-selected routing could be influential in shifting the balance of power (28); we assert that a tool that allows Internet users to circumvent provider-selected filtering and quality degradation has similar value. Indeed, a Perspective Access Network can be used as such a tool, though it could potentially thwart useful price or service discrimination.

Since Perspective Access Networks may allow a user to select the most relevant geolocation, they may provide an opportunity to improve advertising efficiency, offering advertisers an incentive to support the proliferation of Perspective Access Networks. However, advertisers may have reason to oppose deployment of PANs if such deployment means the loss of ability to dominate a local market, and they may also opt to oppose deployment of PANs simply because they do not fully understand the business

implications.

Finally, recall that PAN, unlike the Tor network with which Blossom happens to interoperate, is not designed with anonymity in mind; projection of network location is really all that PAN seeks to achieve. However, this fact may not be enough to prevent Blossom forwarders from eliciting abuse complaints, and the political climate could easily result in the listing of Blossom forwarders on the increasingly preponderant blacklists that have been purportedly established for the purpose of fighting spam. Specialists have often characterized such blacklists as a form of vigilantism, and it is clear that blacklists have been previously used for purposes of questionable merit (55).

6.4 Future Work

PAN affords a plethora of opportunities for future research and development; in this section, we consider some of the possibilities.

For example, it remains to be determined how well PAN interoperates with environments that deliberately restrict access to resources, such as governments censoring the web sites that their citizens could otherwise view. In such a scenario, researchers need to determine how effectively PAN could provide access to blocked resources despite continual discovery and shutdown of PAN forwarders that enable this access. (Without functional steganography, hiding the identities of those who use a PAN may be quite difficult.)

The locality feature of PAN could be used to improve web searches in the Internet today as well as in the increasingly fragmented Internet of the future. To take advantage of locality in PAN, we would need some sort of mechanism capable of

performing targeted web searches. The idea is that it would be interesting to have a "fragmentation-aware" search engine that references content not available in the particular fragment of the Internet in which the client resides.

Equally interesting are the policy questions that arise from having a system like PAN deployed across the Internet. Many enterprises use end-to-end authentication for some of the services they provide in their private networks, but there are a number of popular services that rely upon the assumption that the only hosts that have access to the service are physically on the same LAN or have particular network-layer addresses. Moreover, deployment of PANs in the Internet could threaten the business models of companies providing or depending on geolocation services for anti-fraud resolution, digital rights management, and spam detection. Convincing these parties to move away from network-layer authentication as the basis for their security will be an interesting task.

At its core, PAN is designed to heal fragmentation, which means that Internet users can potentially use perspectives to gain access to resources to which they did not have access previously. However, PAN can be used to provide access to services that use end-to-end authentication mechanisms, and our scheme from Section 3.5 handles services that use network location as a factor in their security assumptions. Intuitively, we are inclined to believe that PAN deployment is morally just and proper when used to provide access to filtered media content otherwise unavailable to dissidents in oppressive regimes, and morally questionable when used to provide unauthorized access to private resources or subscription services. However, the technology itself makes no such distinction.

Another question is whether PAN can be used to resolve namespace arbitrage; proper use of this system could lead to a reduced number of lawsuits related to trademark contention resulting from allocation of resource names. On the other hand, it might cause trademark resolution, as it relates to names of Internet resources, to become a much thornier issue.

While the goals of PAN and the uses of Blossom networks are not the same as the anonymity goals of the Tor network, there is no intrinsic conflict. It might be worth considering extending the Blossom software to make it useful for clients by providing specific access to perspectives as we have described, anonymous access to content in the manner provided by the current Tor network, or some sort of hybrid of anonymity and specificity. This idea has been publically suggested (150), though the details of the implementation remain unspecified. Clearly, as the Tor network expands to include nodes operating inside regimes around the world with different filtering policies, the experience of Tor users will become less predictable in the absence of the ability for users to exercise some control over specificity. In the long run, the issue at hand is not only about avoiding unwanted location-based optimizations in search engine results: as ISPs and lawmakers act to make use of Internet control points, we can expect an increasing disparity among views of the Internet from different locations.

6.5 Closing Remarks

Our proposed design of Perspective Access Networks is motivated by four main objectives:

- enable perspective sharing across the Internet by providing low-latency, topology-

independent access to resources,

- allow locality in naming,
- provide support for the configuration of policies that satisfy the interests of providers of perspective services, and
- promote decentralized management of names and addresses.

The design we propose satisfies these objectives with the aggregation and scalability limitations described in Chapter 5.

Perspective Access Networks provide a convenient means of providing access to otherwise restricted networks and providing end-to-end connectivity to pairs of Internet nodes that are not directly connected to each other. However, PAN technology is not just a means of sustaining some recondite network design principle; it has practical uses in isolating policy decisions from in-band network technology decisions. We have yet to explore the extent to which multiple large-scale independent PAN networks could reasonably coexist. Nonetheless, with recent new threats to Internet consistency (governance disputes, geolocation services, DNS root disputes, and accidental or deliberate censorship of resources), it is worth considering the design and implications of a radically different vision of the Internet—one without a well-defined core, consisting of fragments whose names and address spaces are not ordained hierarchically.

Indeed, Perspective Access Networks address the core of an ongoing tussle surrounding network neutrality. Both policy decisions and technical decisions that result from this tussle will have a profound impact on the future of Internet applications,

commerce, and freedom, and only recently have these issues received public attention. In 2005, the US Supreme Court struck down common carrier requirements for broadband networks, allowing the possibility that network carriers may choose to provide discriminatory access to content (128). Research has shown that providing discriminatory access is in the best interests of individual carriers, though the impact on incentives for deployment of Internet services could be substantial (137). Technology such as deep packet inspection and other advanced filtering techniques have only recently become economically practical (40). The decisions of infrastructure providers and regulators in the months ahead will have long-lasting effects for the Internet.

References

[1] L. Abba, M. Buzzi, D. Pobric, and M. Ianigro. Introducing Transparent Web Caching in a Local Area Network. In *Proceedings of the 26th International Computer Measurement Group Conference*, December 2000.

[2] J. Abley. Hierarchical Anycast for Global Service Distribution. `http://www.isc.org/pubs/tn/isc-tn-2003-1.html`, 2003.

[3] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley. The Design and Implementation of an Intentional Naming System. In *Proceedings of the ACM Symposium on Operating Systems Principles*, pages 186–201, December 1999.

[4] C. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg, and M. Terpstra. Routing Policy Specification Language (RPSL). Internet Engineering Task Force: RFC 2622, June 1999.

[5] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient Overlay Networks. In *Proceedings of the Eighteenth ACM Symposium on Operating Systems Principles*, pages 131–145, Chateau Lake Louise, Banff, AB, Canada, October 2001.

[6] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish. A Layered Naming Architecture for the Internet. In *Proceedings of the ACM Conference of the Special Interest Group on Data Communication (SIGCOMM 2004)*, September 2004.

[7] A. Barbir, R. Penno, R. Chen, M. Hofmann, and H. Orman. An Architecture for Open Pluggable Edge Services (OPES). Internet Engineering Task Force: RFC 3835, August 2004.

[8] G. Berg, I. Davidson, M.-Y. Duan, and G. Paul. Searching for Hidden Messages: Automatic Detection of Steganography. In *Proceedings of the 15th Innovative Applications of AI Conference*, pages 51–56, 2003.

[9] Bertelsmann Foundation. Risk Assessment and Opinions Concerning the Control of Misuse on the Internet. Results of Representative Surveys Conducted in

the United States, Australia, and the Federal Republic of Germany. *Protecting our Children on the Internet: Towards a New Culture of Responsibility*, 2000.

[10] S. Bhattacharjee, K. L. Calvert, and E. W. Zegura. Active Networking and End-to-End Argument. In *Proceedings of the 1997 International Conference on Network Protocols (ICNP '97)*, 1997.

[11] S. Bradner. Almost a Joke. *Network World Weekly*, 9 April 2001.

[12] S. Bradner. Simple Solutions are Often Wrong. *Network World Weekly*, 20 September 2004.

[13] S. Bradner. Just When You Think Telecom Legislation Can't Get Any Worse, It Does. *Network World Weekly*, 21 November 2005.

[14] S. Bradner. Misunderstanding the Fundamentals of Telecom Reform. *Network World Weekly*, 26 September 2005.

[15] S. Bradner, A. Mankin, and J. I. Schiller. A Framework for Purpose-Built Keys (PBK). IETF `draft-bradner-pbk-frame-06`, June 2003.

[16] M. Bright. BT Puts Block on Child Porn Sites. *The Guardian*, 6 June 2004.

[17] Broadband-Testing. Comparing URL Filtering Databases. `http://www.bluecoat.com/downloads/whitepapers/BBT-URL_Coverage.pdf`, June 2005.

[18] B. Carpenter. Architectural Principles of the Internet. Internet Engineering Task Force: RFC 1958, June 1996.

[19] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002)*, December 2002.

[20] R. Chand and P. Felber. A Scalable Protocol for Content-Based Routing in Overlay Networks. In *Proceedings of the IEEE International Symposium on Network Computing and Applications*, April 2003.

[21] A. Chankhunthood, P. Danzig, C. Neerdaels, M. Schwartz, and K. Worrell. A Hierarhical Internet Object Cache. In *USENIX*, Jan 1996.

[22] M. S. Chase and J. C. Mulvenon. You've Got Dissent! Chinese Dissident Use of the Internet and Beijing's Counter-Strategies. RAND Corporation Monograph/Report MR-1543, 2002.

[23] D. R. Cheriton and M. Gritter. TRIAD: A New Next-Generation Internet Architecture. `http://www-dsg.stanford.edu/triad/`, July 2000.

[24] S. Cheshire and M. Krochmal. DNS-Based Service Discovery. IETF `draft-cheshire-dnsext-dns-sd-02`, March 2002.

[25] Cisco Systems. NAT Stateful Failover for Outside-to-Inside and ALG Support: Stateful NAT Phase 2. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios123/123newft/123t/123t_7/gtsnatay.pdf`, 2004.

[26] D. Clark, R. Braden, A. Falk, and V. Pingali. FARA: Reorganizing the Addressing Architecture. *ACM SIGCOMM Computer Communication Review*, pages 313–321, 2003.

[27] D. Clark, K. Sollins, J. Wroclawski, and T. Faber. Addressing Reality: An Architectural Response to Real-World Demands on the Evolving Internet. In *Proceedings of ACM SIGCOMM 2003 FDNA Workshop*, August 2003.

[28] D. Clark, J. Wroclawski, K. Sollins, and R. Braden. Tussle in Cyberspace: Defining Tomorrow's Internet. In *Proceedings of ACM SIGCOMM*, August 2002.

[29] D. D. Clark. The Design Philosophy of the DARPA Internet Protocols. *Computer Communication Review*, 18(4):106–114, August 1988.

[30] D. D. Clark and M. S. Blumenthal. Rethinking the Design of the Internet: The End to End Arguments versus the Brave New World. *ACM Transactions on Internet Technology (TOIT)*, 1(1), August 2001.

[31] E. Cohen and H. Kaplan. Aging Through Cascaded Caches: Performance Issues in the Distribution of Web Content. In *Sigcomm*, 2001.

[32] D. Crocker and P. Hinden. IP Address Encapsulation (IPAE): A Mechanism for Introducing a New IP. IETF Internet Draft, November 1992.

[33] D. Crocker and P. Overell. Augmented BNF for Syntax Specifications: ABNF. Internet Engineering Task Force: RFC 2234, November 1997.

[34] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield. Plutarch: an Argument for Network Pluralism. *ACM SIGCOMM Computer Communication Review*, 33(4):258–266, 2003.

[35] G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a Type III Anonymous Remailer Protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, pages 2–15, May 2002.

[36] G. Danezis, C. Lesniewski-Laas, M. F. Kaashoek, and R. Anderson. Sybil-Resistant DHT Routing. In *Proceedings of the 10th European Symposium On Research In Computer Security*, September 2005.

[37] S. Deering and D. R. Cheriton. Multicast Routing in Datagram Internetworks and Extended LANs. In *Proceedings of ACM Transactions on Computer Systems*, volume 8, pages 85–111, May 1990.

[38] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the Seventh USENIX Security Symposium*, August 2004.

[39] R. Dingledine, N. Mathewson, and P. Syverson. Challenges in Low-Latency Anonymity (DRAFT). US Naval Research Laboratory CHACS Report 5540-625, 2005.

[40] I. Dubrawsky. Firewall Evolution - Deep Packet Inspection. *Security Focus*, 29 July 2003.

[41] European Parliament. Deal on EU Data Retention Law. Press Release, 15 December 2005.

[42] L. Fan, P. Cao, J. Almeida, and A. Broder. Summary Cache: A scalable Wide-area Web Cache Sharing Protocol. In *SIGCOMM*, 1998.

[43] N. Feamster, H. Balakrishnan, and J. Rexford. Some Foundational Problems in Interdomain Routing. In *Proceedings of the Third Workshop on Hot Topics in Networks*, November 2004.

[44] N. Feamster, M. Balazinska, G. Harfst, H. Balakrishnan, and D. Karger. Infranet: Circumventing Censorship and Surveillance. In *Proceedings of the 11th USENIX Security Symposium*, August 2002.

[45] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol: HTTP/1.1. Internet Engineering Task Force: RFC 2616, June 1999.

[46] B. Ford. Unmanaged Internet Protocol. In *Proceedings of the Second Workshop on Hot Topics in Networks*, November 2003.

[47] P. Francis and R. Gummadi. IPNL: A NAT-extended Internet Architecture. In *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 69–80, 2001.

[48] H. W. French. Despite Web Crackdown, Prevailing Winds Are Free. *The New York Times*, 9 February 2006.

[49] R. Frost. Mending Wall. *North of Boston*, 1915.

[50] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. Internet Engineering Task Force: RFC 1519, September 1993.

[51] M. Gaynor and S. Bradner. Firewall Enhancement Protocol (FEP). Internet Engineering Task Force: RFC 3093, April 2001.

[52] M. Girault. Self-Certified Public Keys. *Advances in Cryptology (EUROCRYPT)*, pages 490–497, 1991.

[53] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding Routing Information. In R. Anderson, editor, *Proceedings of Information Hiding: First International Workshop*, pages 137–150. Springer-Verlag, LNCS 1174, May 1996.

[54] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin. Working Around BGP: An Incremental Approach to Improving Security and Accuracy of Interdomain Routing. In *Proceedings of the Network and Distributed System Security Symposium*, February 2003.

[55] P. Graham. The Destiny of Blacklists. `http://paulgraham.com/spamhausblacklist.html`, June 2005.

[56] T. G. Griffin, F. B. Shepherd, and G. Wilfong. Policy Disputes in Path Vector Protocols. In *Proceedings of ICNP*, November 1999.

[57] T. G. Griffin, F. B. Shepherd, and G. Wilfong. The Stable Paths Problem and Interdomain Routing. *IEEE/ACM Transactions on Networking*, 10(1):232–243, 2002.

[58] T. G. Griffin and G. Wilfong. An Analysis of the MED Oscillation Problem in BGP. In *Proceedings of the Tenth International Conference on Network Protocols (ICNP 2002)*, November 2002.

[59] T. G. Griffin and G. Wilfong. On the Correctness of IBGP Configuration. In *Proceedings of SIGCOMM*, August 2002.

[60] M. Gritter and D. R. Cheriton. An Architecture for Content Routing Support in the Internet. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, March 2001.

[61] E. Guttman. Autoconfiguration for IP Networking: Enabling Local Communication. *IEEE Internet Computing*, pages 81–86, June 2001.

[62] E. G. Hallnor and S. K. Reinhardt. A Fully Associative Software-Managed Cache Design. In *Proceedings of the 27th Annual Symposium on Computer Architecture*, June 2000.

[63] A. Harwood and M. Truong. Multi-space Distributed Hash Tables for Multiple Transport Domains. In *Proceedings of the IEEE International Conference on Networks*, pages 283–287, 2003.

[64] L. R. Helfer and G. B. Dinwoodie. Designing Non-National Systems: The Case of the Uniform Domain Name Dispute Resolution Policy. *William and Mary Law Review*, 43(141), 2001.

[65] I. Hilton. When Everything Has Its Price. *The Guardian*, 27 August 1996.

[66] Intel Corporation. The Evolution of the Next-Generation Internet. `http://www.planet-lab.org/`, 2003.

[67] International Telecommunication Union. Network Grade of Service Parameters and Target Values for Circuit-Switched Services in the Evolving ISDN. Recommendation E.721, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, May 1999.

[68] J. Jia and P. Smith. Psiphon: Analysis and Estimation. `http://pyre.third-bit.com/2004-fall/psiphon_ae.html`, October 2004.

[69] Johnny Ngan, et al. Tor Exit Traffic Ports Statistics. Manuscript, 2006.

[70] N. F. Johnson and S. Jajodia. Exploring Steganography: Seeing the Unseen. *IEEE Computer*, pages 26–34, February 1998.

[71] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. In *STOC*, 1997.

[72] J. Kempf and R. Austein. The Rise of the Middle and the Future of End-to-End: Reflections on the Evolution of the Internet Architecture. Internet Engineering Task Force: RFC 3724, March 2004.

[73] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. Internet Engineering Task Force: RFC 2401, November 1998.

[74] S. Kent, C. Lynn, and K. Seo. Design and Analysis of the Secure Border Gateway Protocol (S-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, April 2000.

[75] J. Kristoff. Anycast Addressing on the Internet. *Kuro5hin*, December 2003.

[76] H. T. Kung, C.-M. Cheng, K.-S. Tan, and S. Bradner. Design and Analysis of an IP-Layer Anonymizing Infrastructure. In *Proceedings of the Third DARPA Information Survivability Conference and Exposition (DISCEX 3)*, April 2003.

[77] J.-J. Laffont, S. Marcus, P. Rey, and J. Tirole. Internet Interconnection and the Off-Net-Cost Pricing Principle. *RAND Journal of Economics*, 34(2), 2003.

[78] M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones. SOCKS Protocol Version 5. Internet Engineering Task Force: RFC 1928, March 1996.

[79] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. Wolff. A Brief History of the Internet. `http://www.isoc.org/internet/history/brief.shtml`, 2000.

[80] H. Lewis. Online Fraud Catchers: Protecting You but Maybe Also Getting Your Card Turned Down. `http://www.intelligentbanking.com/brm/news/ob/20000915.asp`, December 2002.

[81] J. Leyden. Verizon Faces Lawsuit over Email Blocking. *The Register*, 21 January 2005.

[82] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP Misconfiguration. In *Proceedings of ACM SIGCOMM 2002*, pages 3–16. ACM, September 2002.

[83] D. Mazieres. *Self-Certifying File System*. PhD thesis, 2000.

[84] D. McCullagh. Court Ponders Web Site Blocking Law. *CNET News.com*, 6 January 2004.

[85] D. McCullagh. U.S. Blunders with Keyword Blacklist. *CNET News.com*, 3 May 2004.

[86] J. Menaud, V. Issarny, and M. Banatre. A New Protocol for Efficient Transversal Web Caching. In *Symposium on Distributed Computing*, 1998.

[87] P. Mockapetris. Domain Names: Concepts and Facilities. Internet Engineering Task Force: RFC 1034, November 1987.

[88] P. Mockapetris and K. Dunlap. Development of the Domain Name System. In *Proceedings of ACM SIGCOMM*, 1987.

[89] Morningstar, Inc. Construction Rules for Morningstar Indexes. `http://indexes.morningstar.com/Index/PDF/Rulebook.pdf`, May 2004.

[90] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host Identity Protocol. IETF `draft-ietf-hip-base-00`, October 2003.

[91] S. Murphy. BGP Security Vulnerabilities Analysis. IETF `draft-ietf-idr-bgp-vuln-00`, February 2002.

[92] NASDAQ. NASDAQ-100 Index Methodology. `http://www.nasdaqtrader.com/trader/defincludes/N100IndexMethod.pdf`, 2006.

[93] National Research Council. Global Networks and Local Values. National Academy Press, 2001.

[94] N. Netanel. Cyberspace Self-Governance. *Law, Information, and Information Technology*, 2001.

[95] A. Newey. Freedom of Expression. *Liberating Cyberspace: Civil Liberties, Human Rights, and the Internet*, 1999.

[96] T. S. E. Ng, I. Stoica, and H. Zhang. A Waypoint Service Approach to Connect Heterogeneous Internet Address Spaces. In *Proceedings of the USENIX Annual Technical Conference 2001*, June 2001.

[97] W. B. Norton. Internet Service Providers and Peering. In *Proceedings of NANOG 19*, May 2001.

[98] P. O'Connell. At SBC, It's All About "Scale and Scope". BusinessWeek Online, 7 November 2005.

[99] Open Net Initiative. Internet Filtering in China 2004-2005. Open Net Initiative Case Study, June 2005.

[100] Open Net Initiative. Internet Filtering in Iran 2004-2005. Open Net Initiative Case Study, June 2005.

[101] Open Net Initiative. The Internet and Elections: the 2006 Presidential Election in Belarus. Open Net Initiative Case Study, April 2006.

[102] J. G. Palfrey. Stemming the International Tide of Spam. *Trends in Telecommunication Reform 2006*, 7:111–125, March 2006.

[103] C. Partridge, T. Mendez, and W. Milliken. Host Anycasting Service. Internet Engineering Task Force: RFC 1546, November 1993.

[104] C. Perkins. IP Mobility Support for IPv4. Internet Engineering Task Force: RFC 3220, January 2002.

[105] H. Petersen and P. Horster. Self-Certified Keys: Concepts and Applications. In *Proceedings of the Third International Conference on Communications and Multimedia Security*, pages 102–116, 1997.

[106] R. Power. 2002 CSI/FBI Computer Crime and Security Survey. *Computer Security Issues and Trends*, 8(1), 2002.

[107] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proceedings of the ACM Conference of the Special Interest Group on Data Communication (SIGCOMM)*, pages 161–172, August 2001.

[108] M. G. Reed, P. F. Syverson, and D. M. Goldschlag. Proxies for Anonymous Routing. In *Proceedings of the 12th Annual Computer Security Applications Conference*, pages 95–104, December 1996.

[109] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP 4). Internet Engineering Task Force: RFC 1771, March 1995.

[110] Reuters. Egyptologists Launching Online Encyclopedia. `http://www.cnn.com/2006/TECH/science/04/27/egyptology.online.reut/index.html`, 27 April 2006.

[111] C. Rhoads. Endangered Domain. *The Wall Street Journal*, 19 January 2006.

[112] V. Roto and A. Oulasvirta. Need for Non-Visual Feedback with Long Response Times in Mobile HCI. In *In Proceedings of WWW2005*, May 2005.

[113] A. Salkever. Web Worms Can Google, Too. BusinessWeek Online, July 2004.

[114] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-End Arguments in System Design. *ACM TOCS*, 2(4):277–288, November 1984.

[115] H. Schulzrinne and K. Arabshian. Providing Emergency Services in Internet Telephony. *IEEE Internet Computing*, pages 39–45, May 2002.

[116] L. Sherriff. So why does Vodafone Filter Block Sky News? *The Register*, 6 July 2004.

[117] U. Sieber. Responsibility of Internet Providers. *Law, Information, and Information Technology*, 2001.

[118] P. Smith. Internet Routing Table Analysis Update. Presented at SANOG 7, Mumbai, India, January 2006.

[119] A. C. Snoeren and H. Balakrishnan. An End-to-End Approach to Host Mobility. In *Proceedings of ACM/IEEE MOBICOM'99*, August 1999.

[120] A. C. Snoeren, H. Balakrishnan, and M. F. Kaashoek. Reconsidering Internet Mobility. In *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, May 2001.

[121] A. C. Snoeren and B. Raghavan. Decoupling Policy from Mechanism in Internet Routing. In *Proceedings of the Second Workshop on Hot Topics in Networks*, November 2003.

[122] M. Srivatsa and L. Liu. Vulnerabilities and Security Threats in Structured Overlay Networks: a Quantitative Analysis. In *20th Annual Computer Security Applications Conference*, December 2004.

[123] S. Staniford, V. Paxson, and N. Weaver. How to 0wn the Internet in Your Spare Time. In *Proceedings of the 11th USENIX Security Symposium*, August 2002.

[124] J. Stewart. BGP4: Interdomain Routing in the Internet. Addison-Wesley, 1998.

[125] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana. Internet Indirection Infrastructure. In *Proceedings of ACM SIGCOMM*, August 2002.

[126] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *SIGCOMM*, 2001.

[127] T. E. Sundsted. Restoring the Transparent Network, Part 1. *IBM developerWorks*, August 2002.

[128] Supreme Court of the US. FCC v. Brand X Internet Services et al. 04-281, March 2005.

[129] D. Talbot. The Internet is Broken. *Technology Review*, 108(11), December 2005.

[130] R. Tewari, M. Dahlin, H. Vin, and J. Kay. Design Considerations for Distributed Caching on the Internet. In *ICDCS*, 1999.

[131] J. Touch. The LSAM Proxy Cache - A Multicast Distributed Virtual Cache. In *3rd WWW Caching Workshop*, Jun 1998.

[132] United States Public Laws. Gramm-Leach-Billey Act. 106 PL 102, 113 Stat 1338, 119 Enacted S.900, November 1999.

[133] United States Public Laws. Sarbanes-Oxley Act. PL 107-204, 116 Stat 745, 2002 Enacted HR.3763, 2002.

[134] UPnP Forum. UPnP Device Architecture. `http://www.upnp.org/download/UPnPDA10_20000613.htm`, 2000.

[135] US Court of Appeals. US versus Bradford C. Councilman. 03-1383, June 2004.

[136] V. Valloppilli and K. Ross. Cache Array Routing Protocol v1.0. IETF `draft-vinod-carp-v1-02.txt`, Feb 1998.

[137] B. Van Schewick. Towards an Economic Framework for Network Neutrality Regulation. In *Proceedings of the Telecommunications Policy Research Conference*, September 2005.

[138] K. Varadhan, R. Govindan, and D. Estrin. Persistent Route Oscillations in Inter-Domain Routing. *Computer Networks*, 32(1):1–16, 2000.

[139] C. Villamizar, R. Chandra, and R. Govindan. BGP Route Flap Damping. Internet Engineering Task Force: RFC 2439, November 1998.

[140] N. Villeneuve. The Filtering Matrix: Integrated Mechanisms of Information Control and the Demarcation of Borders in Cyberspace. *First Monday*, 11(1), January 2006.

[141] M. Walfish, H. Balakrishnan, and S. Shenker. Untangling the Web from DNS. In *Proceedings of the USENIX/ACM Symposium on Networked Systems Design and Implementation*, March 2004.

[142] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker. Middleboxes No Longer Considered Harmful. OSDI, 2004.

[143] D. Wessels. Squid Internet Object Cache. `http://www.squid-cache.org/`, 2001.

[144] A. Williams. Requirements for Automatic Configuration of IP Hosts. IETF `draft-ietf-zeroconf-reqts-12`, March 2002.

[145] World Wide Web Consortium. HTML 4.01 Specification. `http://www.w3c.org/TR/html401/`, December 1999.

[146] T. Wright. EU Tries to Unblock Internet Impasse. *The New York Times*, 30 September 2005.

[147] B. Y. Zhao, Y. Duan, L. Huang, A. D. Joseph, and J. D. Kubiatowicz. Brocade: Landmark Routing on Overlay Networks. In *Proceedings of the International Workshop on Peer-to-Peer Systems*, March 2002.

[148] S. Q. Zhuang, K. Lai, I. Stoica, R. H. Katz, and S. Shenker. Host Mobility using an Internet Indirection Infrastructure. In *Proceedings of the First International Conference on Mobile Systems, Applications, and Services*, May 2003.

[149] J. Zittrain. The Generative Internet. *Harvard Law Review*, 119, May 2006.

[150] E. Zuckerman. Blossom, Tor and Touring the Internets. `http://www.ethanzuckerman.com/blog/?p=473`, 6 April 2006.