Network Working Group
Internet-Draft
Intended status: Informational
Expires: May 21, 2009

R. Alimi
Yale University
D. Pasko
Verizon
L. Popkin
Pando Networks, Inc.
Y. Wang
Y. Yang, Ed.
Yale University

November 17, 2008

P4P: Provider Portal for P2P Applications
draft-p4p-framework-00.txt

Status of This Memo

Abstract

P4P is a framework that enables Internet Service Providers (ISPs) and network application software distributors to work jointly and cooperatively to optimize application communications. The goals of this cooperation are to reduce network resource consumption and to accelerate network applications. In this document, we specify how P4P allows ISPs to provide network guidance to applications, allowing

clients to exchange data more effectively. The applications we focus on are those that can have a choice in connection patterns, in particular peer-to-peer (P2P) applications.

Table of Contents

1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

P4P, which stands for provider portal for network applications, is a framework that enables Internet Service Providers (ISPs) and network application software developers to work jointly and cooperatively to optimize application communications. The goals of this cooperation are to reduce network resource consumption and to accelerate applications. To achieve these goals, P4P allows ISPs to provide network information and guidance to network applications, allowing clients to exchange data more effectively. In this specification, we focus on peer-to-peer (P2P) applications.

The P4P framework was initially developed in the P4P Working Group (P4PWG), hosted by the Distributed Computing Industry Association (DCIA). Many members made contributions to the development and discussion of the framework. The authors of this document recorded the technical development. The complete list of contributors is listed in Appendix A.

The P4P framework has been validated by both simulations and field trials from February to August 2008. The goal of this document is to record the technical approach implemented in those trials, and to be a first step in contributing to the production of an eventual industry standard (e.g., in the ALTO working group). This document focuses on the framework, basic concepts, and major features of P4P. Exact protocol specification will be specified in another document, but we give several example messages in Appendix C. Most features presented in this document have been implemented and tested in field tests. There are revisions on the interfaces and their presentations, based on field test experiences, to improve the framework.

The specific goals of the P4P include the following:

- Allows P2P networks to select the "close" peers for data exchanges, such that data transit cost/distance is minimized and data transfer speed is maximized;

- Is extensible and applicable to a wide range of P2P applications including file sharing, multimedia streams, multi-player gaming, auditorium/chat systems, etc. These applications may either include a centralized component (e.g., BitTorrent with trackers,

Pando) or be distributed (e.g., Gnutella).

- Allows each ISP to manage P4P guidance that is provided for its customers.

- Does not specify how ISPs generate their P4P guidance. A configuration language can be used by ISPs to configure their networks, as we did in our field tests. However, this is outside the scope of this document.

- ISPs can control the degree of the privacy of information revealed about their networks. ISPs can determine the level of detail that they wish to expose, and apply any desired aggregation and allocation metrics, and all information can be "anonymized" or access-controlled to protect ISP topology and business policies.

- Does not specify how P2P networks utilize the information that ISPs provided. Our field tests gave a reference implementation scheme, but this is outside the scope of this document.

- P2P networks can choose schemes to avoid exposing user-specific information to ISPs.

- Allows "peers" to include a range of data sources, such as standard peers operated by end-users, dedicated "P2P cache servers" operated by ISPs, etc.

- Does not specify how multiple ISPs interact or negotiate with respect to their information. This was an interesting issue during the field tests, but it is outside the scope of this document.

3. P4P Terminology and Entities

This document involves the following terms and entities.

- P2P Client: A P2P client, or Peer or Client for short, is run by a user, and creates network connections to endpoints (e.g., clients or servers) in its ISP's network or other ISPs' networks.

- Network Location Identifier: It is either an IP address, an IP prefix or an autonomous system number (ASN).

- PID: A Partition ID, or a PID for short, is an identifier for a set of Network Location Identifiers (see Section 6.1 for more discussion). It is a generalization of network aggregation concepts such as autonomous system (AS) or OSPF area. It can denote a subnet, a set of subnets, a point of presence (PoP), an

Internet exchange point, an autonomous system (AS), a set of autonomous systems (e.g., those with the same next-hop AS, those reachable through provider AS's, or those with the same BGP AS_PATH length), or some other set of Clients with a common set of properties. PIDs provide three primary functions:

* Mapping and aggregation of Network Location Identifiers;

* Preserving privacy by avoiding identification of individual Clients or specific details of network topology; and

* Identifying network resources.

Each ISP partitions the Internet address space with a set of PIDs that it defines. This is referred to as the my-Internet view of the ISP (see Section 9.2.3 for more discussion). To differentiate the name space of PIDs from different ISPs, each ISP MUST be identified by a universal identifier (e.g., a unique number assigned by IANA or a URI such as isp1.openp4p.info). Such an identifier is named an ISP identifier. Since many ISPs have multiple ASNs, the PIDs defined within a particular ISP identifier may span multiple ASNs.

o pDistances: An ISP network reveals its information and preferences through generalized distances between pairs of Network Location Identifiers or aggregation of Network Location Identifiers (i.e., PIDs). We refer to such distances generically as pDistances. pDistances MAY have attributes.

  * There MAY exist an attribute indicating the Type of pDistances. Example Types include Routing Hop-Count pDistances, Routing Air-Mile pDistances, and Routing Cost pDistances; see Section 9.1.1 for more discussion. ISPs MUST support Routing Cost pDistances, which are computed internally according to ISPs local state and policies. The Routing Cost pDistances are extensions to path metrics computed by OSPF traffic engineering routing, multihoming optimization, and BGP to unify intradomain routing and interdomain routing to integrate P2P applications [SIGCOMM08]; see Section 9.1.1 for detail. In the absence of the Type attribute, applications SHOULD assume that any pDistances given are Routing Cost pDistances.

  * There MAY exist an attribute indicating whether a given set of pDistances should be interpreted as either numerical or ordinal (ranking, i.e., the best has a pDistance value of 1, next 2, etc; see [Oracle]). Certain arithmetic operations (e.g., summation) may not be meaningful with ordinal pDistances. The choice of using numerical PID-pair pDistances as the main

interface is based on rigorous research derivations [SIGCOMM08]. There are also recent studies (e.g., [NetEcon08]) supporting the completeness of such an interface design. In the absence of this attribute, applications SHOULD assume that given pDistances are numerical.

- ISP Portal Service: Each ISP or its delegate provides Portal Services to answer P4P queries [SIGCOMM08]. We refer to the (logical) entity implementing a Portal Service a Portal Server. In the field tests, we also referred to a Portal Server as an iTracker. In this document, we focus on two Portal Services: (1) Location Portal Service and (2) pDistance Portal Service. Multiple Portal Services may run on the same physical machine. A (logical) service may also be implemented using multiple physical machines either in a cluster or an overlay. These physical machines may be interconnected using a protocol (e.g., forming a hierarchy) that is outside the scope of this document. As an example, in our field tests, some Portal Services were implemented using LinuxHA by multiple physical machines for fault tolerance. A Portal Service may be extended by a P2P system to improve scalability.

- Location Portal Service: This service answers queries concerning mappings between Network Location Identifiers and PIDs.

- pDistance Portal Service: This service answers queries about pDistance between PIDs.

- Application Tracker: Many P2P applications use Application Trackers for bootstrapping and coordinating the connections among Clients. Application Trackers may use, among other data, information from Portal Services, to guide or make suggestions to Clients. We also referred to an Application Tracker as an AppTracker or pTracker in the field tests. Note that an Application Tracker is not used by all applications and is an OPTIONAL component of the system.

- Application Optimization Engine (AppOE): The Application Optimization Engine is a service introduced in P4P field tests where a (possibly external) entity converts ISP information into a format more directly applicable to the specific application's decision process. An Application Optimization Engine provides three benefits: (1) modular integration of P4P and P2P applications, (2) allowing a P2P application to offload this functionality to a third party, and (3) providing an aggregation point for better management and access control of ISP information. Note that this is an OPTIONAL component of the system, since ISP information may be applied using various other methods.

4. P4P Interfaces

4.1. Overview

To handle diverse P2P application scenarios, the P4P framework adopts an interface-centric design. The design supports heterogenous uses such as tracker-based and tracker-less P2P applications, and also allows applications to integrate P4P information either directly or through a relatively independent module such as an Application Optimization Service. This document focuses on ISP-guided initial Peer selection.

The basic P4P information flow for P2P Peer selection involves two services:

- The Location Portal Service to map between PIDs and Network Location Identifiers;
- The pDistance Portal Service to map the pDistances between network locations.

We separate these two services because (1) they provide different functions; and (2) the location information may be large but stable for a much longer time-scale than the pDistance information.

With pDistances, an application makes its Peer selection decision by considering not only ISP pDistances but also other data such as application state and policies.

Below, we specify more details on the interfaces. The interfaces provided by the Portal Services are discussed. After specifying these interfaces, in the next two sections we present example usages. Example details are given in Appendix C.

4.2. Location Portal Service

A P4P supporting ISP MUST offer a Location Portal Service that supports mapping between PIDs and Network Location Identifiers.

4.2.1. Interfaces

The Location Portal Service provides two interfaces.

The first interface is named GetPID. In this interface, the Location Portal Service directly performs the mapping from Network Location Identifiers to PIDs. This interface MUST be implemented.

- GetPID: This interface takes a list of Network Location Identifiers and returns the corresponding list of PIDs. If the requester supplies an empty list in the request, the reply returns the PID corresponding to the source IP address of the sender's request.

The second interface is named GetPIDMap. This interface provides information to applications so that they can perform the mapping from Network Location Identifiers to PIDs themselves. There are three advantages with this interface: (1) reducing the querying load on ISP Portal Service; (2) reducing application latency; and (3) protecting individual Client privacy. This interface SHOULD be implemented.

- GetPIDMap: This interface returns a mapping from PIDs to lists of Network Location Identifiers. If an empty list of PIDs is specified in the request, the reply MAY include the mappings for PIDs that can define Routing Cost pDistances. If a list of PIDs is specified in the request, the reply returns only the mappings for those PIDs or a subset of those PIDs.

4.2.2. Querying Entities

The Location Portal Service may be queried by any entity requiring mappings between PIDs and Network Location Identifiers. We name two possibilities here.

- A Client may directly query its Location Portal Service through the GetPID interface. The PID may then be sent to other endpoints with which the Client is communicating, such as other Clients or an Application Tracker (if one is used). See Section 9.2.2 for further discussion.

- An Application Tracker or another entity (AppOE) may query the PID of a Client, either due to a design consideration or because of incremental deployment (e.g., delegate queries for unmodified Clients).

4.3. pDistance Portal Service

A P4P supporting ISP MUST offer a pDistance Portal Service.

4.3.1. Interface

The pDistance Portal Service answers queries concerning pDistances between Network Location Identifers and PIDs.

This service MUST provide the following interface:

- GetpDistances: This interface takes as input a list of PID->PID pairs and desired attributes (e.g., Type) of the pDistances. The reply is a list of PID->PID pairs (it may be a subset of those supplied in the request) and their associated pDistances. The requester may specify pairs of Network Location Identifiers instead of pairs of PIDs in the request. In such a case, the reply will be pDistances for the specified pairs of Network Location Identifiers.

5. Example Usage of P4P Interfaces

The P4P interfaces may be used in a variety of ways by ISPs and applications depending on their particular behavior and requirements. In this section, we provide examples of how the interfaces can be queried in some typical scenarios.

5.1. Example E1: Querying for Individual Client Addresses

One way to provide guidance to applications is to issue a request for each new Client.

Specifically, in this usage case, when a new Client of an ISP joins, the Client or Application Tracker issues a GetpDistances query for the IP address (or IP prefix) of the Client and a list of IP addresses (or IP prefixes) of candidate Peers.

The application then selects Peers internally by attempting to meet application-specific requirements and utilizing the returned pDistances to account for ISP guidance.

5.2. Example E2: Aggregated Query Using PIDs

An issue of the preceding usage scenario is that it may impose significant load on the Application Tracker (if there is one and it performs the query) and the ISP pDistance Portal. Privacy is another concern, since the IP address of the Client is directly linked to other Clients' IP addresses and they are possibly downloading the same content since they are queried together. Due to these issues, the preceding usage scenario was not used in our field tests. Instead, an application can use PIDs for aggregation. PIDs allow ISPs to aggregate "like" Clients, for example, Clients in the same metropolitan area. We consider an example when there is an Application Tracker.

An Application Tracker keeps track of Clients in the system. With P4P, the Application Tracker will also need to maintain ISP information. If there exists a Client that has a given PID-p of ISP-i, we say that PID-i is active in ISP-i. Consider an

implementation where the Application Tracker maintains two additional data structures for each ISP: (1) the pDistance table for the pDistances among those active PIDs of the ISP; and (2) the Clients at each active PID of the ISP. The second data structure helps the Application Tracker to quickly select Peers from a given PID.

When a new Client, Client-a, from ISP-i joins but does not report its PID, the Application Tracker looks up the PID of Client-a (e.g., either using GetPID or looking up in the local PID Map if the Application Tracker has called GetPIDMap for ISP-i). If the PID of Client-a is already active, the Application Tracker can make Peer selection for Client-a utilizing the current pDistance table (or derived guidance structure such as a peering weight data structure); otherwise, the Application Tracker queries the pDistance Portal of ISP-i to extend its pDistances table for ISP-i to include the new PID. Note that for Client-a to be selectable as a Peer when a Client from another ISP joins, the PIDs of Client-a for other ISPs need to be resolved, and inserted into other ISPs' second data structure. This can be done in a background process using a queue, and may be done for only a subset of ISPs other than ISP-i to improve scalability.

5.3. Example E3: Utilizing an Application Optimization Service

The Application Optimization Service is a technique we introduced in P4P to convert ISP pDistances disseminated by ISP Portal Services into a format more-directly applicable for guiding application behavior. The intentions of introducing this service are: (1) achieving modular integration of P4P and P2P applications; (2) allowing a P2P application to offload this functionality to a third party; and (3) providing an information aggregation point for ISPs to control the distribution of their information.

It is important to note that the Application Optimization Service utilizes the previously-defined ISP Interfaces. Application Optimization Services could be provided by ISPs as an additional interface (to augment the mandatory interfaces), by third-parties, or integrated as a component of a P2P application.

Since the Application Optimization Service provides guidance specific to an application or class of applications, the information and format will vary depending on the application.

The following diagram shows how an Application Optimization Service may be used to transform ISP pDistances into application-specific information.

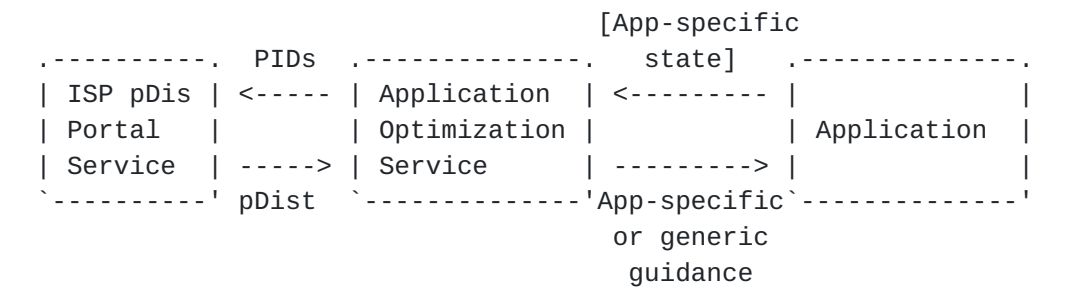

Figure 1: Information flow when there is Application Optimization Service.

Note that an Application Optimization Service may provide generic guidance that does not make use of application state information. In such a case, an application may simply request generic guidance without supplying application-specific state information. Note that generic guidance may still depend on the type of application (e.g., file sharing vs streaming). In our field tests for file sharing, we achieved good performance with generic guidance.

Since the format of the guidance provided from the Application Optimization Service is particular to the type of application (file sharing and streaming may choose peering differently), next we give an example of using an Application Optimization Service for file sharing.

In the field tests, an Application Optimization Service for P2P file-sharing applications is implemented. It defines this simple interface:

- GetPeeringWeights: The request optionally includes swarm state information as a list of PIDs, and for each PID, the number of seeds and leechers and the aggregated download and upload capacity of clients within the PID. The response is a matrix of peering weights amongst the PIDs included in the request, as computed from the set of pDistances currently pulled from Portal Servers. If the request included swarm information, the returned weight matrix is tailored for the current state of the swarm. Otherwise, the returned matrix is generic guidance that can be used across different swarm states.

The Application Optimization Service implemented for the field test periodically retrieved updated pDistances from ISP pDistance Portal Services.

6. Extended Usage of PIDs

In this section we illustrate how the flexibility of PIDs caters to more diverse usage scenarios by both applications and ISPs. Such usage scenarios are not tested in previous tests. However, they do not require any modification to the communication protocol. In particular, we show how PIDs support caches and simple avoid/prefer lists.

We first give more details on PIDs. Then we provide usage scenarios.

6.1. PID Type

Although PIDs are always used for aggregation, we identify that they may aggregate different types of objects. The preceding section uses PIDs to aggregate network locations. It is also possible to use a PID to aggregate a set of network resources. We call the first type Aggregation PIDs and the second Resource PIDs.

6.1.1. Aggregation PIDs

Aggregation PIDs are used to group "like" clients (identified by Network Location Identifiers) into PIDs. Aggregation PIDs allow an ISP to perform customized aggregation of Network Location Identifiers for its network and locations external to its network.

Note that there is a well-known PID named PID_ISP_DEFAULT which implicitly contains all Network Location Identifiers not contained by other Aggregation PIDs. Mapping of an IP address to a set of Aggregation PIDs is always based on the most specific matching. See Section 9.1.3 for more details.

6.1.2. Resource PIDs

A Resource PID is used to identify a particular available resource (e.g., caches) to applications. The objects aggregated by a Resource PID may be Endpoint Identifiers (an IP address and associated attributes such as port number, protocol, authorization, etc). Thus, when Resource PIDs are enumerated by the Location Portal Interface, each returned PID should be mapped to a list of Endpoint Identifiers.

This framework defines some specific Resource PIDs with well-known semantics, but others may be allocated by an organization such as the IANA.

6.2. Example X1: Using PIDs to Identify Caches

Some Resource PIDs are used to identify caches available in an ISP. An application uses GetPIDMap on the following Resource PIDs to obtain caches in an ISP network:

- PID_RES_PUBLIC_CACHE: enumerates a set of caches publicly available in the ISP network;
- PID_RES_PRIVATE_CACHE: enumerates a set of caches, but demands authorization before doing so.

After an application obtains a list of Endpoint Identifiers from the GetPIDMap interface for a Resource PID identifying caches, it can then get pDistances to these endpoints by mapping the IP addresses of the identified caches into Aggregation PIDs and then querying the GetpDistance interface. There is an additional approach to obtain caches. Please see Section 9.1.5.

Note that additional Resource PIDs may be defined for caches understanding specific protocols. For example, there can be Resource PIDs that identify caches for BitTorrent.

6.3. Example X2: Using PIDs for ISP AVOID/PREFER

An ISP may wish to reveal some or all of its preferences in the form of preferring or avoiding specific Network Location Identifiers. The usage of avoid/prefer is motivated and similar to the proposal in [p2pi-Shalunov]. As an example, consider an ISP network with two types of customers. The first type (type I) may have limited upload capacity. Thus, the ISP may prefer that the first type of customers do not generate a large amount of upload traffic to others.

One possible way to reflect this preference is the following. The ISP configures the Location Portal Service to define an Aggregation PID named PID_ISP_AVOID in addition to PID_ISP_DEFAULT:

- PID_ISP_AVOID: avoided Network Location Identifiers.

Then, Clients with Network Location Identifiers defined in PID_ISP_AVOID will be given PID_ISP_AVOID, and those outside will be given PID_ISP_DEFAULT.

The ISP configures its Routing Cost pDistances as those in Figure 2, where MIN_PDISTANCE and MAX_PDISTANCE are the minimum and maximum pDistance values respectively. In particular, upload from the avoided sources (i.e., type I) is configured with the maximum pDistance while upload from the preferred sources (i.e., the other

type) is configured with the minimum pDistance.

		Destination	
		PID_ISP_DEFAULT	PID_ISP_AVOID
Source	PID_ISP_DEFAULT	MIN_PDISTANCE	MIN_PDISTANCE
	PID_ISP_AVOID	MAX_PDISTANCE	MAX_PDISTANCE

Figure 2: pDistances for implementing ISP AVOID/PREFER.

An ISP can extend beyond two levels of preferences and configure multiple tiers using a similar approach.

7. Portal Server Discovery

The field tests used manual configuration for the discovery of Location and pDistance Servers. In the P4P framework design, possibilities include DNS SRV (e.g., BEP 22). As another possibility, Portal Servers can be found through a P4P DNS hierarchy. For example, for a Client with IP address a.b.c.d to find its Portal Servers, it may query for d.c.b.a.openp4p.org, where openp4p.org conducts mapping from IP addresses to their Portal Servers.

A key issue in discovering the Portal Servers is delegation. In our field tests, multiple ISPs proposed the possibility of delegation: an ISP provides information for costumer networks that may not want to run Portal Servers themselves. An ideal solution is that such networks have their DNS entries pointing to the delegation Portal Servers. Thus, a DNS based solution may be more ideal.

8. Security Considerations

There are security considerations from the perspectives of both ISPs and P2P applications.

8.1. Security Considerations for ISPs

- ISPs MAY wish to implement access control to some or all P4P interfaces to only trusted entities. This can be achieved, for example, by running the interfaces on top of TLS/SSL. However, the exact mechanism for access control, authentication, and confidentiality of message transport is outside the scope of this document.

- ISPs need to evaluate how much information to reveal and the associated risks. For example, revealing extremely fine-grained information may make it easier to determine network topology while revealing overly coarse-grained information may not provide benefits to the ISP or to applications. Also, a malicious attacker may target those PID-pairs with high pDistances. To alleviate these risks, ISPs may apply several techniques, including (1) aggregation (e.g., a single PID for intradomain to block internal topology and reveal only interdomain preferences); (2) perturbation of revealed information; and (3) access control (see above). A related comment is that some information that the pDistance Portal reveals is not secret information. For example, information based on hop-count and air-miles is largely measurable by Clients themselves (e.g., when there is no MPLS).

- Another risk that ISPs need to evaluate is that some other ISPs may distribute information leading to less desirable traffic patterns. In the P4P architecture, the preference of an ISP is applied only when the new client is from the ISP. This distributes the control of ISPs. However, one can still envision scenarios where an ISP may have a sufficient number of Clients where guidance can affect another ISP.

8.2. Security Considerations for P2P Applications

P2P applications may encounter ISP behaviors that would be considered hostile from their perspective. Consider the following examples.

- The PIDs may be fine-grained. Although querying the interfaces does not link to any particular content, it may help ISPs to track Clients. One potential technique to alleviate this issue is that the application truncates IP addresses. Another technique is to use only GetPIDMap.

- There is a risk that ISPs could provide ineffective guidance. For example, the network pDistances (either Routing related pDistances or Routing Cost pDistances) configured by an ISP may lead to lower application performance. To address this issue, it is important that applications are robust and detect such behaviors, possibly through community efforts. Applications may still use other mechanisms to complement ISP guidance or replace ISP guidance when it is ineffective.

- Some ISP Portals may be poorly provisioned or even intentionally under-provisioned, leading to substantial delay. Applications should be designed to tolerate failure of ISP portals. For example, in our field tests, ISP information is cached.

9. Discussion and Extensions

There are many considerations that contributed to the design of these P4P interfaces. This section further elaborates on some items.

9.1. ISP Information

9.1.1. pDistance Semantics and Calculation

Motivated by OSPF, which provides multiple types of link metrics, we allow multiple Types of pDistances. It is important that the semantics of a given Type of pDistance is as well defined as possible. For example, Routing Hop-Count pDistance, Routing Air-Mile pDistance, and Routing Cost pDistance represent the number of hops, the air milage, and the traffic engineering cost of transmitting one bit from a given source network location to a given destination network location, respectively.

However, application developers should be aware that there is inherent approximation when an ISP computes and reveals such information. We discuss two illustrative examples.

Routing Hop-Count pDistances: given a pair of Internet hosts, the routing hop count from a source to a destination is typically well-defined. However, the pDistance computed by the ISP may be an approximation. There are multiple reasons.

- First, when the ISP provides pDistance between a pair of PIDs instead of end hosts, it may approximate the hop count pDistance as the average hop count between end hosts within those PIDs. For potentially better accuracy, the pDistance Portal allows queries using Network Location Identifiers. However, the ISP may still choose to internally map Network Location Identifiers into PIDs and then return pDistances.

- Second, if a PID represents a location outside the ISP, the ISP may need to merge its intradomain and interdomain routing information. Thus, the hop count from a PID representing a location inside the ISP to a PID representing a location outside the ISP, may be the sum of intradomain hop count and interdomain BGP hop count. In our early design, two metric spaces were considered: one metric space for the pDistances among PIDs inside the ISP, and the other metric space for the pDistances from PIDs inside the ISP to PIDs outside the ISP. The pDistances from these two metric spaces are not comparable. However, multiple scenarios were identified where an ISP may prefer some interdomain connections over some intradomain connections. In the current design, a single metric space to define pDistances allows ISPs to

define a uniform policy.

Routing Cost pDistance: In traditional Internet OSPF traffic engineering, an ISP computes effective OSPF link costs to improve routing efficiency [OSPF-TE]. The P4P Routing Cost pDistances are extensions to traditional OSPF traffic engineering costs to integrate P2P applications. However, different ISPs may have different traffic engineering cost metrics. Thus, there is inherent fuzziness when defining Routing Cost pDistances. In the July 2008 field test, we found that the ISPs approximated their Routing Cost pDistances in a variety of ways, including air-miles, hop counts, and OSPF-derived costs. An interesting observation from our studies is that there was strong positive correlation between ISP-configured pDistances and geographic distances.

9.1.2. pDistance Direction

Also motivated by OSPF, we allow pDistance to be asymmetric: the pDistance from PID_1 to PID_2 can be different from PID_2 to PID_1. One issue to consider when using such information for Peer selection, however, is that it is in general hard to predict the traffic direction between a pair of Clients. Another issue is that the routing system of an ISP may not have accurate information from a PID outside its network to a PID inside its network.

9.1.3. Aggregation PIDs

There are subtle issues involved in defining Aggregation PIDs. We discuss some of them.

The Aggregation PIDs are particularly important for interdomain. For example, instead of specifying hundreds of thousands of IP prefixes in the global Internet, or tens of thousands of autonomous systems, Aggregation PIDs may allow an ISP to specify pDistances for a much smaller set of objects. Note that such aggregation will payoff only if the mapping from Clients to PIDs can be obtained at a low cost (e.g., through a database without involving the ISP), or the mapping is more stable and the pDistances are more frequently queried.

Aggregation may depend on the location inside an ISP network. This is particularly so for BGP. For example, the intention of an Aggregation PID may be a set that includes all external IP destinations using a given interdomain exchange point. This set may depend on the source location inside the ISP network. Thus, an external destination AS may have multiple exist points. Defining a PID for each destination with multiple exist points may require defining too many PIDs. An ISP may apply technique (e.g., choosing only the top or randomization) to reduce the number of such PIDs.

We allow an ASN to be a Network Location Identifier for scalability and convenience. An ASN typically is used to identify an aggregated location outside of an ISP. The BGP tables of the routers of an ISP define association between an IP address and its origin ASN. An issue arises, for example, when an Application Tracker uses PID maps and the definition of a PID includes ASNs. Since the Application Tracker does not have access to the BGP tables of the ISP, it needs a method to map from IP addresses to ASNs. Thus, if a PID includes ASNs, the ISP may need to ensure that applications can achieve reasonably accurate mapping from IP addresses to the used ASNs. In our field tests, an Application Tracker used a public database to conduct this mapping. Note that using this database may introduce errors.

There is a possibility that two PIDs defined by an ISP overlap; that is, an IP address belongs to the Network Location Identifiers of both PIDs. For the purpose of mapping this IP address to a PID, we assume that ASN has precedence over IP prefix. Among IP prefixes, the longest prefix match takes precedence. If two ASNs from two different PIDs contain the IP address, one of them is picked arbitrarily.

9.1.4. Hierarchical PIDs

It is possible to impose a hierarchical structure on PIDs to improve scalability and allow delegation among Portal Servers (e.g., when defining pDistances). For example, an ISP, isp1, may assign a Client within its network a PID such as subpid1.pid1.intra.isp1, which denotes that the Client belongs to subpid1, which is a part of pid1, which is a part of the internal network of isp1. A Client outside isp1 may be assigned a PID such as asn100.pid_exit_point.exter.isp1, which denotes that the external Client is seen as originated from ASN 100, and will take the exit point denoted by pid_exit_point. This hierarchical format allows easier checking on whether two Clients are within the same hierarchy. However, this is outside the scope of this document.

9.1.5. PID Attributes

During our early field tests, some ISPs were able to provide additional information such as Client access type (e.g., ADSL 1Mbps down/384kbps up) and geographical location (to a certain precision). One may also query the caches that are close to a PID. As another example, some ISPs provide geo-location of PIDs.

Such information may be encoded as the attributes of a PID. To support queries on such information, we can add a new interface:

- GetPIDAttribute, which takes as input a PID and the desired attribute type, and returns the value of the attribute.

9.2. Scalability

Scalability of the interfaces is a major design consideration. We discuss several items.

9.2.1. Caching P4P Information

It is recommended that the reply from Portal Services should include a lifetime attribute to facilitate caching. This is not provided in the current specification as it was not implemented in the field tests.

9.2.2. Client PID Retrieval

To improve scalability, it is suggested that Clients directly query their Location Portal Service through the GetPID interface for their PIDs either at startup, or when first needed. Since a PID is application-agnostic and content-independent, it may be cached for a period of time within the Client and used across applications and communicated with other endpoints.

9.2.3. Global PIDs and my-Internet View

A major scalability challenge of using ISP information by a P2P application is that the application needs to handle the "my-Internet" views of multiple ISPs. The my-Internet view concept is strongly prefered by some ISPs in configuring their networks during our field tests.

In applications where Clients are concentrated in a few ISPs, such multiple views may not be an issue. Also, the application may choose to handle only the top ISPs to reduce overhead. Furthermore, the application may not need to always resolve the PID of Client outside its home ISP.

From our experiences, specifying interdomain information contributes to a large fraction of ISP specification, as interdomain may involve hundreds of thousands of IP prefixes and/or tens of thousands of autonomous systems. Globally consistent PIDs (e.g., based on ideas such as [GeoIDRouting07]) when defining PIDs for locations outside an ISP may improve scalability. Such an approach may also allow PID maps from different ISPs to be linked, for example, at peering or exchange points. However, this requires coordination among the "views" of different ISPs.

9.2.4. Granularity of Information

ISPs should be mindful of overloading applications with overly-detailed data. In one possible extreme case, an ISP could configure its Location Portal Service to define a large number of PIDs (e.g., one for each CMTS or DSLAM).

Applications, however, may need to store data from multiple Portal Servers. If the provided data is extremely fine-grained, an application may not have the resources to store or process such data. In such a case, the application can revert to ignoring ISP-provided guidance. Since ISP-provided guidance can benefit the ISP, there are incentives to provide P4P information compact enough such that applications may store and process it, yet also conveying the desired preferences.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[GeoIDRouting07] R. Oliveira, M. Lad, B. Zhang, L. Zhang, "Geographically Informed Inter-domain Routing", In ICNP 2007.

[NetEcon08] P. Laskowski, B. Johnson, and J. Chuang, "User-Directed Routing: From Theory, towards Practice", In ACM NetEcon 2008.

[OSPF-TE] B. Fortz and M. Thorup, "Internet traffic engineering by optimizing OSPF weights", In IEEE INFOCOM 2000.

[Oracle] Vinay Aggarwal, Anja Feldmann, Christian Scheideler, "Can ISPs and P2P systems co-operate for improved performance?", In CCR 2007.

[SIGCOMM08] H. Xie, Y.R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz., "P4P: Provider Portal for (P2P) Applications", In ACM SIGCOMM. 2008.

[p2pi-Shalunov] S. Shalunov. In p2pi discussion list: http://www.ietf.org/mail-archive/web/p2pi/current/msg00508.html, "ALTO service privacy, performance,

and architecture", 2008.

Appendix A. Contributors

The P4P project includes contributions from many members of the P4P Working Group, hosted by DCIA.

The individuals involved in the design and P4P field tests include (in alphabetical order):

- Richard Alimi, Yale University
- Alex Gerber, AT&T
- Chris Griffiths, Comcast
- Ramit Hora, Pando Networks
- Arvind Krishnamurthy, University of Washington
- Y. Grace Liu, IBM Watson
- Jason Livingood, Comcast
- Michael Merritt, AT&T
- Doug Pasko, Verizon
- Laird Popkin, Pando Networks
- James Royalty, Pando Networks
- Thomas Scholl, AT&T
- Emilio Sepulveda, Telefonica
- Avi Silberschatz, Yale
- Hassan Sipra, Bell Canada
- Haibin Song, Huawei
- Oliver Spatscheck, AT&T
- Jia Wang, AT&T
- Richard Woundy, Comcast

- Hao Wang, Yale University
- Ye Wang, Yale University
- Haiyong Xie, Yale University
- Y. Richard Yang, Yale University

Appendix B. Acknowledgments

The authors would like to thank the members of the P4P Working Group for their collaboration, and the members of the p2pi mailing list for their comments and questions. We would like to think Marty Lafferty from DCIA, Erran Li, Jin Li, See-Mong Tang, and Yu-Shun Wang for reading the document and giving us excellent feedback.

Appendix C. P4P Protocol Example

C.1. Overview

In this section we provide example message flows to illustrate how P2P applications may request P4P information from ISP Portal Services and how P4P information can be used. Note that the examples below are not traces from our field tests, but rather are constructed to be illustrative. Our field test implementation used WSDL to specify P4P interfaces, and SOAP as the encoding. Exact messaging format will be presented in a spec document. Below, we give interface invocations without giving the encoding. In the example, we use strings as PIDs. We also do not include the ISP identifier since we consider a single ISP.

We begin with an example similar to the tracker-based example in Section 5.2. We also include a variant for tracker-less systems.

C.2. ISP Portal Service Configuration

An ISP configures its Location Portal Service to maintain the mapping of Network Location Identifiers to PIDs, and configures the pDistances between each pair of PIDs in the pDistance Portal Service.

In this example, the ISP defines five Aggregation PIDs in addition to PID_ISP_DEFAULT. Three of these PIDs represent intradomain IP addresses, PID_EAST, PID_WEST, and PID_MIDDLE, which relate to their geographic locations. These three PIDs will be marked as INTRA. The remaining two PIDs represent interdomain endpoints: PID_EX_EAST is configured with IP addresses and ASNs reached through peering links in the east; PID_EX_WEST is configured with those reached through peering links in the west. They will be marked as EXTER.

The ISP prefers traffic to remain in the same geographic area, so it configures the pDistance from PID_EAST to PID_EAST to be 0, and similarly for PID_WEST and PID_MIDDLE. To express its preference for intradomain over interdomain traffic, the remaining pDistances amongst the intradomain PIDs PID_EAST, PID_WEST, and PID_MIDDLE are configured smaller than pDistances between intradomain PIDs and the interdomain PIDs PID_EX_EAST and PID_EX_WEST.

C.3. Tracker-based P2P Application

In a tracker-based file-sharing P2P application, the Application Tracker maintains the swarm state. In this example, it retrieves P4P information directly from the ISP's Portal Service. Finally, it uses the information to optimize the initial Peer selection.

1. The Application Tracker queries the ISP's Location Portal Service to retrieve the PID map:

   GetPIDMap Request

   Parameter: none (indicating a request for a map of all PIDs defined by the ISP).

   GetPIDMap Response

   ```
   PID_ISP_DEFAULT 0/0
   PID_EAST INTRA 128.36.0.0/16
   PID_MIDDLE INTRA 216.8.0.0/16
   PID_WEST INTRA 206.0.0.0/8 209.234.0.0/16
   PID_EX_EAST EXTER AS294 77.0.0.0/8 93.0.0.0/8
   PID_EX_WEST EXTER AS4571 AS4981 112.0.0.0/8 126.0.0.0/8
   ```

2. Six Clients (Peers) join the swarm. Each Peer reports its IP address to the Application Tracker. Application Tracker locally determines each Peer's PID from the PID map.

   ```
   Client 128.36.233.132:  PID_EAST
   Client 112.72.31.251:   PID_EX_WEST
   Client 206.8.179.24:    PID_WEST
   Client 93.132.128.199:  PID_EX_EAST
   Client 128.36.233.98:   PID_EAST
   Client 126.199.253.7:   PID_EX_WEST
   ```

3. The Application Tracker queries the pDistance Portal Service for pDistances. Only the active PIDs (see Section 5.2) are specified in the request. Note that the response does not contain the full pDistance matrix.

GetpDistance Request

Parameter:

```
PID_EAST PID_EAST
PID_EAST PID_WEST
PID_EAST PID_EX_WEST
PID_EAST PID_EX_EAST
PID_WEST PID_EAST
PID_WEST PID_WEST
PID_WEST PID_EX_WEST
PID_WEST PID_EX_EAST
PID_EX_WEST PID_EAST
PID_EX_WEST PID_WEST
PID_EX_EAST PID_EAST
PID_EX_EAST PID_WEST
```

GetpDistance Response

```
PID_EAST PID_EAST 0
PID_EAST PID_WEST 15
PID_EAST PID_EX_WEST 140
PID_EAST PID_EX_EAST 75
PID_WEST PID_EAST 16
PID_WEST PID_WEST 0
PID_WEST PID_EX_WEST 92
PID_WEST PID_EX_EAST 128
PID_EX_WEST PID_EAST 140
PID_EX_WEST PID_WEST 92
PID_EX_EAST PID_EAST 75
PID_EX_EAST PID_WEST 128
```

4. When a Client at PID_WEST or PID_EAST requests a set of Peers from the Application Tracker, the Application Tracker determines the PID for the requesting Peer, and constructs a Peer list for the Client.

C.4. Tracker-less P2P Application

We now present a simple example for a tracker-less P2P application. This approach may be used for tracker-less P2P protocols, or for cases where an Application Tracker does not support P4P.

1. A Client begins by querying the Location Portal Service's GetPID interface at startup (see Section 9.2.2) to find its PID.

GetPID Request

Parameter: none (indicating a request for the client's PID).

GetPID Response

PID_EAST INTRA

2. After the Client obtains a Peer list (e.g., from a DHT or gossiping), it queries the Location Portal Service to find the PIDs of the Peers in the list. The GetPID request now includes a list of the IP addresses of potential Peers. IP addresses are truncated to increase privacy.

GetPID Request

Parameter:

128.36.233.0
112.72.31.0
206.8.179.0
93.132.128.0
128.36.233.0
126.199.253.0

GetPID Response

128.36.233.0 PID_EAST INTRA
112.72.31.0 PID_EX_WEST EXTER
206.8.179.0 PID_WEST INTRA
93.132.128.0 PID_EX_EAST EXTER
128.36.233.0 PID_EAST INTRA
126.199.253.0 PID_EX_WEST EXTER

3. The Client queries the pDistance Portal Service to determine the pDistances between itself and the Peers in the list. The Client supplies its own PID together with other Peers' PIDs in the GetpDistance request.

GetpDistance Request

Parameter:

PID_EAST PID_EAST
PID_EAST PID_WEST
PID_EAST PID_EX_EAST
PID_EAST PID_EX_WEST

GetpDistance Response

```
PID_EAST PID_EAST 0
PID_EAST PID_WEST 15
PID_EAST PID_EX_EAST 75
PID_EAST PID_EX_WEST 140
```

4. The Client then prefers Peers whose PIDs have smaller pDistances.

5. When the Client receives a new Peer list (for example, through gossiping), it queries GetPID to map the newly-discovered Peers to PIDs, and obtains pDistances if necessary.

Authors' Addresses

Richard Alimi
Yale University

EMail: richard.alimi@yale.edu

Doug Pasko
Verizon

EMail: doug.pasko@verizon.com

Laird Popkin
Pando Networks, Inc.

EMail: laird@pando.com

Ye Wang
Yale University

EMail: ye.wang@yale.edu

Y. Richard Yang (editor)
Yale University

EMail: yry@cs.yale.edu

Full Copyright Statement

Intellectual Property