ALTO WG R. Penno, Ed.
Internet-Draft Juniper Networks
Intended status: Standards Track Y. Yang, Ed.
Expires: January 14, 2010 Yale University
July 13, 2009

ALTO Protocol
draft-penno-alto-protocol-03.txt

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on January 14, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

Applications already have access to great amount of underlying network topology information. For example, views of the Internet

routing table are easily available at looking glass servers and entirely practical to downloaded by clients. What is missing is network side information such as the network preference information -- what an ISP or Content Provider actually prefers -- and a way to distribute it.

The ALTO Service provides information such as preferences of network resources with the goal of modifying network resource consumption patterns while maintaining or improving application performance. This document describes a protocol implementing the ALTO Service. While such service would primarily be provided by the network (i.e., the ISP), content providers and third parties could also operate this service. Applications that could use this service are those that have a choice in connection endpoints. Examples of such applications are peer-to-peer (P2P) and content delivery networks.

Requirements Language

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 RFC 2119 [1].

Table of Contents

1. Introduction

1.1. Background and Problem Statement

Today, network information available to applications is mostly from the view of endhosts. There is no clear mechanism to convey information about the network's preferences to applications. By leveraging better network-provided information, applications have potential to become more network-efficient (e.g., reduce network resource consumption) and achieve better application performance (e.g., accelerated download rate). The ALTO Service intends to provide a simple way to convey network information to applications.

The goal of the protocol specified in this document is to provide a simple, unified protocol that meets the ALTO requirements [5],

providing a migration path for Internet Service Providers (ISP), Content Providers, and clients that have deployed protocols with similar intentions (see below). This document is a work in progress and will be updated with further developments.

1.2. Design History and Merged Proposals

The protocol specified here consists of contributions from

- o P4P [6],[7];
- o ALTO Info-Export [8];
- o Query/Response [9],[10];
- o ATTP [ATTP].
- o Proxidor [19].

The people listed here should be viewed as co-authors of this document: Obi Akonjang, Richard Alimi, Saumitra M. Das, Syon Ding, Anja Feldmann, Doug Pasko, Reinaldo Penno, Laird Popkin, Stefano Previdi, Satish Raghunath, Stanislav Shalunov, Albert Tian, Yu-Shun Wang, Richard Woundy, Y. Richard Yang, David Zhang, and Yunfei Zhang. Due to the limit of 5 authors per draft, the complete list of authors were moved to the contributors section at this point.

1.3. Solution Benefits

The ALTO Service offers many benefits to both end-users (consumers of the service) and Internet Service Providers (providers of the service).

1.3.1. Service Providers

The ALTO Service enables ISPs to influence the neighborhood selection process of overlay networks to increase locality of traffic and also regain the ability to efficiently engineer traffic that traverses more expensive links such as backbone and transit links, thus allowing a better provisioning of the networking infrastructure.

1.3.2. P2P Applications

Applications that use the ALTO Service can benefit in multiple ways. For example, they may no longer need to infer topology information, and some applications can reduce reliance on measuring path performance metrics themselves. They can take advantage of the ISP's knowledge to avoid bottlenecks and boost performance.

2. Architecture

Two key design objectives of the ALTO Protocol are simplicity and extensibility. At the same time, it introduces additional techniques to address potential scalability and privacy issues. Below we start with an introduction to the terminology. Then we define the overall architecture and how the ALTO Protocol fits into the architecture. At the end of this section, we specify the simple, but general protocol framework which demonstrates its extensibility.

2.1. Terminology

We use the following terms defined in [11]: Application, Overlay Network, Peer, Resource, Resource Identifier, Resource Provider, Resource Consumer, Resource Directory, Transport Address, Host Location Attribute, ALTO Service, ALTO Server, ALTO Client, ALTO Query, ALTO Reply, ALTO Transaction, Local Traffic, Peering Traffic, Transit Traffic.

We also use the following additional terms: Endpoint Address and Network Location.

2.1.1. Endpoint Address

An endpoint address represents the communication address of an end point. An endpoint address can be network-attachment based (IP address) or network-attachment agnostic. Common forms of endpoint addresses include IP address, MAC address, overlay ID, and phone number.

2.1.2. Network Location

Network Location is a generic concept denoting a single endpoint or group of endpoints. Whenever we say Network Location, we refer to either a single endpoint or a group of endpoints.

2.2. ALTO Service and Protocol Scope

An ALTO Server conveys the network information from the perspective of a network region. We say that the ALTO Server presents its "my-Internet View" [12] of the network region. A network region in this context can be an Autonomous System, an ISP, perhaps a smaller region, or perhaps a set of ISPs; the details depend on the deployment scenario and discovery mechanism.

To better understand the ALTO Service and the role of the ALTO Protocol, we show in Figure 1 the overall system architecture. In this architecture, an ALTO Client uses ALTO Service Discovery to identify an appropriate ALTO Server; an ALTO Server prepares ALTO Information; and the ALTO Client requests available ALTO Information from the ALTO Server using the ALTO Protocol.

The ALTO Information provided by the ALTO Server can be updated dynamically based on network conditions, or can be seen as a policy which is updated at a larger time-scale.

More specifically, the ALTO Information provided by an ALTO Server may be influenced (at the operator's discretion) by other systems. Examples include (but are not limited to) static network configuration databases, dynamic network information, routing protocols, provisioning policies, and interfaces to outside parties. These components are shown in the figure for completeness but outside the scope of this specification.

```
   +-------------------------------------------------------------------+
   |                               ISP                                 |
   |                                                                   |
   |                    +-----------+                                  |
   |                    | Routing   |                                  |
   |  +--------------+  | Protocols |                                  |
   |  | Provisioning |  +-----------+                                  |
   |  | Policy       |        |                                        |
   |  +--------------+\       |                                        |
   |                   \      |                                        |
   |                    \     |                                        |
   |  +-----------+      \+---------+                      +--------+  |
   |  |Dynamic    |       | ALTO    | ALTO Protocol        | ALTO   |  |
   |  |Network    |.......| Server  | -------------------- | Client |  |
   |  |Information|       +---------+                      +--------+  |
   |  +-----------+      /                                /            |
   |                    /        ALTO SD Query/Response /              |
   |                   /                               /               |
   |          +----------+                 +--------------+            |
   |          | External |                 | ALTO Service |            |
   |          | Interface|                 | Discovery    |            |
   |          +----------+                 +--------------+            |
   |               |                                                   |
   |               |         Figure 1: Basic ALTO Architecture.       |
   |               |                                                   |
   +-------------------------------------------------------------------+
                   |
        +------------------+
        | Third Parties    |
        |                  |
        | Content Providers|
        +------------------+
```

ALTO Architecture

2.3. Query Types

As a general framework, ALTO Information is provided via the ALTO Protocol by answering the following four types of queries:

2.3.1. Server Capability

It lists the details on the information that can be provided by an ALTO Server.

2.3.2. Endpoint Property

Given an endpoint, it gives the set of network-aware properties associated with the endpoint. An example endpoint property is its Network Location property or connectivity type (e.g., ADSL, Cable, or FioS).

2.3.3. Reverse Property Map

It is the reverse of the preceding. In particular, given a property, it lists the endpoints with the property.

2.3.4. Path Property Lookup

It gives information on the properties of paths among Network Locations.

3. Network Map

The preceding section specifies a simple, extensible ALTO Protocol framework. In this section, we focus on a particular endpoint property named Network Map. In the next section, we introduce a particular path property named Path Rating.

In reality many endpoints are very close to one another in terms of network connectivity, for example, endpoints on the same site of an enterprise. By treating a group of endpoints together as a single entity in ALTO, we can achieve much greater scalability without loosing any critical information.

The Network Location endpoint property allows an ALTO Server to group endpoints together to indicate their proximity. The resulting set of groupings is called the ALTO Network Map.

The Network Map may also be used to communicate simple preferences. For example, an ISP may prefer that endpoints associated with the same PoP (Point-of-Presence) in a P2P application communicate locally instead of communicating with endpoints in other PoPs.

Note that the definition of proximity varies depending on the granularity of the ALTO algorithm. In one deployment, endpoints on the same subnet may be considered close; while in another deployment, endpoints connected to the same PoP may be considered close.

3.1. PID

Each group can be identified by a Network Location identifier called a PID. There can be many different ways of grouping the endpoints and assigning PIDs.

Thus, a PID is an identifier providing an indirect and network-agnostic way to specify a network aggregation. For example, a PID may be defined (by the ALTO service provider) to denote a subnet, a set of subnets, a metropolitan area, a PoP, an autonomous system, or a set of autonomous systems. Aggregation of endpoints into PIDs can indicate proximity. Also, aggregation can improve scalability. In particular, network preferences (costs) may be specified between PIDs, allowing cost information to be more compact and updated at a smaller time scale than the network aggregations themselves.

3.2. Example Network Map

Figure 1 illustrates an example Network Map. PIDs are used to identify network-agnostic aggregations.

```
.---------------------------------------------------------.
| ALTO Network Map                                        |
|                                                         |
|  .----------------------------------.  .---------------. |
|  | NetLoc: PID-1                    |  | NetLoc: PID-2 | |
|  |  .----------------------------.  |  |     ...       | |
|  |  | 128.36.0.0/16              |  |  `---------------` |
|  |  | .------------------------. |  |                    |
|  |  | | Endpoint: 128.36.9.8   | |  |  .---------------. |
|  |  | `------------------------` |  |  | NetLoc: PID-3 | |
|  |  `----------------------------`  |  |     ...       | |
```

```
   |   |   .----------------------------.  |   `---------------`  |
   |   |   | 130.132.0.0/16             |  |                      |
   |   |   | .------------------------. |  |   .---------------.  |
   |   |   | | Endpoint: 130.132.1.2  | |  |   | NetLoc: PID-4 |  |
   |   |   | `------------------------` |  |   |     ...       |  |
   |   |   `----------------------------`  |   `---------------`  |
   |   `-------------------------------`                          |
   |                                                              |
   `--------------------------------------------------------------`
```

Figure 1: Example Network Map

3.3. Endpoint PID Query

The Endpoint Property query against the Network Map allows an ALTO Client to retrieve the PID of a given endpoint.

3.4. Reverse Network Map Query

The Reverse Property Map query for Network Map allows an ALTO Client to obtain a map from PIDs to lists of endpoints: for each PID, the map includes its list of endpoints.

4. Path Rating

In this section we define a particular path property named Path Rating.

4.1. Path Cost

Path Rating is based on Path Cost, which conveys the preference of an ALTO Server on communication among Network Locations. Path Costs have attributes:

o Type: identifies what the costs represent;

o Mode: identifies how the costs should be interpreted (numerical or ordinal interpretation).

4.1.1. Cost Type

The Type attribute indicates what the cost represents. For example, an ALTO Server could define costs representing air-miles, hop-counts, or generic routing costs.

Cost types are indicated in protocol messages as alphanumeric strings. An ALTO Server MUST at least define the routing cost type denoted by the string 'routingcost'.

Note that an ISP may internally compute routing cost using any method it chooses (including air-miles or hop-count).

If an ALTO Client requests a Cost Type that is not available, the ALTO Server responds with an error as specified in Section 6.2.1.2.

4.1.2. Cost Mode

The Mode attribute indicates how costs should be interpreted. For example, an ALTO Server could return costs that should be interpreted

as numerical values or ordinal rankings.

It is important to communicate such information to ALTO Clients, as certain operations may not be valid on certain costs returned by an ALTO Server. For example, it is possible for an ALTO Server to return a set of IP addresses with costs indicating a ranking of the IP addresses. Arithmetic operations, such as summation, that would make sense for numerical values, do not make sense for ordinal rankings. ALTO Clients may want to handle such costs differently.

Cost Modes are indicated in protocol messages as alphanumeric strings. An ALTO Server MUST at least define the modes 'numerical' and 'ordinal'.

If an ALTO Client requests a cost Mode that is not supported, the ALTO Server MUST reply with costs having Mode either 'numerical' or 'ordinal'. Thus, an ALTO Server must implement at least one of 'numerical' or 'ordinal' Costs, but it may choose which to support.

ALTO Clients may choose how to handle such situations. Two particular possibilities are to use the returned costs as-is (e.g., treat numerical costs as ordinal rankings) or ignore the ALTO information altogether.

4.2. Path Rating Query

The Path Rating Query consists of a Cost Type, a Cost Mode, a list of Source Network Locations (note that a Network Location can be an endpoint address or a PID), and a list of Destination Network Locations.

Specifically, assume that a Path Rating query has a list of multiple Source Network Locations, say [Src_1, Src_2, ..., Src_m], and a list of multiple Destination Network Locations, say [Dst_1, Dst_2, ..., Dst_n], then the ALTO Server will compute the Path Cost for each communicating pair (i.e., Src_1 -> Dst_1, ..., Src_1 -> Dst_n, ..., Src_m -> Dst_1, ..., Src_m -> Dst_n).

4.2.1. Cost Map

We refer to the Response containing the m*n entries as a Cost Map.

If the Cost Type is ordinal, the ranking of each communicating pair is relative to the m*n entries.

4.2.2. Ranking List

If there is a single Source Network Location and the Cost Mode is ordinal, we also say that the response is a Ranking List.

4.2.3. Implicit Source Network Location

If the Source Network Location is not specified in the Query message, it is inferred by the ALTO server as the address of the Query message sender.

4.2.4. Implicit Destination Network Location

If the Destination Network Location is not specified in the Query message, it is inferred by the ALTO server as the list of all PIDs.

4.2.5. Network Map and Cost Map Dependency

Note that if a Path Rating query contains any PID in the list of Source Network Locations or the list of Destination Network Locations, we say that the particular Path Rating is generated based on a particular Network Map. Version Tags are introduced to ensure that ALTO Clients are able to use consistent information even though the information is provided in two maps. One advantage of separating ALTO information into a Network Map and a Cost Map is that the two components can be updated at different time scales. For example, Network Maps may be stable for a longer time while Cost Maps may be updated to reflect dynamic network conditions.

5. Protocol Overview

5.1. Design Approach

The ALTO Protocol design uses a RESTful interface using a lightweight XML encoding, with the goal of leveraging current HTTP [2] [3] implementations and infrastructure. ALTO messages are denoted with HTTP Content-Type "application/alto".

These design decisions make the protocol easy to understand and debug, and allows for flexible ALTO Server implementation strategies. More importantly, however, this enables use of existing implementations and infrastructure, and allows for simple caching and redistribution of ALTO information to increase scalability.

5.1.1. Use of Existing Infrastructure

An important design consideration for the ALTO Protocol is easy integration with existing applications and infrastructure. As outlined above, HTTP is a natural choice. In particular, this ALTO Protocol design leverages:

- o the huge installed base of HTTP infrastructure, including HTTP caches,
- o mature software implementations for both HTTP and XML,

o the fact that many P2P clients already have an embedded HTTP client, and

o authentication and encryption mechanisms in HTTP and SSL.

5.1.2. ALTO Information Reuse and Redistribution

ALTO information may be useful to a large number of applications and users. Distributing ALTO information must be efficient and not become a bottleneck. Therefore, the ALTO Protocol specified in this document integrates with existing HTTP caching infrastructure to allow reuse of ALTO information by ALTO Clients and reduce load on ALTO servers. ALTO information may also be cached or redistributed using application-dependent mechanisms, such as P2P DHTs or P2P file-sharing.

For example, a Cost Map amongst PIDs may be reused by all ALTO Clients within a particular Source Grouping [12]. A full Network Map may be reused by all ALTO Clients using the ALTO Server.

5.2. Message Format

The ALTO Protocol uses a RESTful design operating over HTTP. Both Query and Response follow the standard format for HTTP Request and Response messages [2] [3]. This section provides an overview of the components of a Query message sent from an ALTO Client to an ALTO Server, as well as the components of a Response message returned by an ALTO Server. Note that if caching or redistribution is used, the Response message may be returned from another (possibly third-party) entity. Reuse and Redistrubution is further discussed in Section 10.4.

5.2.1. Query Message

A Query message is generated by an ALTO Client and sent to an ALTO Server. The ALTO Protocol employs the following components of the HTTP request message:

Method: Indicates operation requested by the ALTO Client (along with URI Path).

URI Path: Indicates the operation requested by the ALTO Client (along with Method).

URI Query String Parameters: Indicates parameters for the requested operation. Note that in the messaging specification in Section 6, we abbreviate these as 'URI QS Params'. Order of query string parameters is not specified. Some parameters are restricted in how many times they appear. We use the notation 'min..max' to denote the the minimum and maximum times they may appear, where 'max' may be '*' to denote unbounded.

Headers: Indicates encoding of the Body.

Body: Indicates additional request parameters that are not concisely representable as Query String parameters.

5.2.2. Response Message

A Response message is generated by an ALTO Server, which corresponds to a particular Query message. The ALTO Protocol employs the following components of the HTTP Response message:

Status Code: Indicates either success or an error condition.

Headers: Indicates encoding of the Body and caching directives.

Body: Contains data requested by the ALTO Client.

5.2.3. Query and Response Body Encoding

When the Body of a Query or Response message is not empty, it MUST contain a well-formed XML document and it SHOULD contain an encoding declaration in the XML declaration. If the charset parameter of the MIME content type declaration is present and it is different from the encoding declaration, the charset parameter takes precedence. Every application conforment to this specification MUST accept the UTF-8 character encoding to ensure maximum interoperability. The namespace for the elements defined in this specification is urn:ietf:params:xml:ns:p2p:alto.

```
<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
    ...
</alto>
```

Example XML Document Carried by ALTO Protocol Messages

6. Protocol Messaging

This section specifies client and server processing, as well as messages in the ALTO Protocol. Details common to ALTO Server processing of all messages is first discussed, followed by details of the individual messages. Note that the primary focus of the current draft is the architecture and protocol operations. This section will be updated as revisions are made to protocol details and encodings. For clarity of the examples, details such as URL encoding have been omitted.

6.1. Client Processing

6.1.1. General Processing

An ALTO Client implementing the ALTO protocol can make use of a set of queries, each for a different purpose. The protocol is structured in such a way that independent of the query type there are a set of general processing steps. The ALTO Client selects a specific ALTO Server to communicate with and establishes a TCP connection. The ALTO protocol on top of this TCP connection can be secured through SSL/TLS. The client then decides which query to use and formats it as specified in Section 6.3, which includes HTTP request-line, headers and body. Finally the client sends it down the TCP/IP stack. All HTTP encoding rules apply, as well as TCP transport considerarions.

6.1.2. General Error Conditions

In the case the client does not receive an appropriate response from the server it can choose another server to communicate or fall back to perform peer selection without the use of ALTO information.

6.2. Server Processing

6.2.1. General Error Conditions

This section specifies ALTO Server behavior when it recevies a Query message that cannot be processed due to a problem with processing the Query message itself.

6.2.1.1. Invalid Query Format

If any component of the Query message is formatted incorrectly (i.e., it does not follow the formats in Section 6.3), the ALTO Server MUST return HTTP Status Code 400.

6.2.1.2. Unsupported Query

If an ALTO Server does not support the operation indicated in the Query message, the ALTO Server MUST return HTTP Status Code 501.

6.3. ALTO Queries

6.3.1. Server Capability

The Server Capability query allows an ALTO Client to determine the configuration of a particular ALTO Server. The configuration includes, for example, details about the operations and cost metrics supported by the ALTO Server. The returned document can be downloaded by ALTO Clients or provisioned into devices.

6.3.1.1. Query

The Query message MUST follow:

```
Method         : 'GET'
URI Path       : '/capability'
URI QS Params  : MUST be empty
Headers        : None required
Body           : MUST be empty
```

6.3.1.2. Response

The Response message MUST follow:

```
Status Code    : '200'
Headers        : 'Content-Encoding: application/alto'
Body           : See Below
```

The Body MUST be an XML document containing the Server Configuration XML structure in Appendix B.1.

6.3.1.3. Example Query and Response

```
GET /capability HTTP/1.1
Host: alto.example.com

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]
```


```
<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <configuration instance-name="alto.example.com">
    <alto-server uri="http://alto.example.com:6671"/>
    <cost type="latency" units="ms"/>
    <cost type="pDistance" units="scalar"/>
    <cost type="bandwidth" units ="kbps"/>
    <constraint-support value="false"/>
  </configuration>
</alto>
```


6.3.2. Endpoint Property Lookup

The Endpoint Property Lookup query allows an ALTO Client to query for properties of Endpoints known to the ALTO Server.

6.3.2.1. Query

There are multiple forms of the Query message. The Query message from the ALTO Client MUST follow one of the forms.

The first form allows an ALTO Client to query for properties of a single endpoint:

```
Method         : 'GET'
URI Path       : '/endpoint/[endpointname]'
URI QS Params  : 'prop=[propertyname]'   (multiplicity: 1..*)
Headers        : None Required
Body           : MUST be empty
```

Note that the '[endpointname]' and '[propertyname]' strings above are placeholders to be substituted with valid values indicated in Appendix A.1 and Appendix A.3, respectively.

Also note that the 'prop' parameter may be specified multiple times to query for multiple properties simultaneously. For example, the query string could be 'prop=pid&prop=bandwidth'.

The second form allows an ALTO Client to query for properties of multiple endpoints:

```
Method         : 'POST'
URI Path       : '/endpoint/m'
URI QS Params  : 'prop=[propertyname]'      (multiplicity: 1..*)
Headers        : 'Content-Encoding: application/alto'
Body           : See Below
```

In the second form, the Body MUST be an XML document containing the Endpoint List XML structure in Appendix B.3, with the additional requirement that 'endpoint' elements specify no attributes except for 'name'.

6.3.2.2. Response

The Response message MUST follow:

```
Status Code    : '200' if all property types are supported
```

```
                      '501' if at least one property is not supported
    Headers         : 'Content-Encoding: application/alto'
    Body            : See Below
```

The Body MUST be an XML document containing the Endpoint List XML structure in Appendix B.3.

6.3.2.3. Example Query and Response

For additional message examples, see Appendix C.1.

```
GET /endpoint/ipv4:128.36.1.34?prop=pid HTTP/1.1
Host: alto.example.com

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <endpoint name="ipv4:128.36.1.34" pid="PID1"/>
</alto>
```

Example Query for Single Endpoint

6.3.3. Reverse Property Lookup

The Reverse Property Lookup query allows an ALTO Client to query for Endpoints with common properties. This draft focuses on the case where an ALTO Client wishes to determine the Endpoints within a PID. For scalability, the Endpoints within a PID may be enumerated using IP Prefixes.

6.3.3.1. Query

There are multiple forms of the Query message. The Query message from the ALTO Client MUST follow one of the forms.

The first form allows an ALTO Client to query for the IP prefixes within a specific PID defined by the ALTO Server:

```
Method         : 'GET'
URI Path       : '/prop/pid/[pidname]'
URI QS Params  : MUST be empty
Headers        : None Required
Body           : MUST be empty
```

Note that the '[pidname]' string above is a placeholder to be substituted with valid values indicated in Appendix A.2.

The second form allows an ALTO Client to query for the IP prefixes within all PIDs defined by the ALTO Server:

```
Method         : 'GET'
URI Path       : '/prop/pid'
URI QS Params  : MUST be empty
Headers        : None Required
Body           : MUST be empty
```

The third form allows an ALTO Client to query for the IP prefixes within a specific set of PIDs:

```
Method         : 'POST'
URI Path       : '/prop/pid/m'
URI QS Params  : MUST be empty
Headers        : 'Content-Encoding: application/alto'
Body           : See Below
```

In the second form, the Body MUST be an XML document containing the PID List XML structure in Appendix B.5, with the additional requirement that 'pid' elements specify no attributes except for 'name'.

6.3.3.2. Response

The Response message MUST follow:

```
Status Code    : '200' if all PIDs specified in request are
```

                             valid, or no PIDs are specified in request.
                       '404' if at least one PID specified in request
                             is not valid.
     Headers   : 'Content-Encoding: application/alto'
     Body   : See Below

The Body MUST be an XML document containing the PID List XML structure in Appendix B.5.

6.3.3.3. Example Query and Response

For additional message examples, see Appendix C.2.

```
GET /prop/pid/PID1 HTTP/1.1
Host: alto.example.com

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <pid name="PID1" size="3">
    <cidr4 name="128.36.1.0/24"/>
    <cidr4 name="132.130.1.0/24"/>
    <cidr4 name="132.130.2.0/24"/>
  </pid>
</alto>
```

Example Query for Single PID

6.3.4. Path Rating Lookup

The Path Rating Lookup query allows ALTO Clients to query for Costs amongst Network Locations.

6.3.4.1. Query

There are multiple forms of the Query message. The Query message from the ALTO Client MUST follow one of the forms.

The first form allows an ALTO Client to query for costs amongst all PIDs defined by the ALTO Server:

```
Method         : 'GET'
URI Path       : '/cost/map'
URI QS Params  : 'type=[costtype]'              (multiplicity: 0..1)
                 'mode=[costmode]'              (multiplicity: 0..1)
                 'constraint=[constraint]'      (multiplicity: 0..*)
Headers        : None Required
Body           : MUST be empty
```

Note that the '[costtype]', '[costmode]', '[constraint]' strings above are placeholders to be substituted with valid values indicated in Appendix A.5, Appendix A.6, and Appendix A.7 respectively.

The 'constraint' parameter is optional and is to be used only if the ALTO service supports it. It allows a client to specify a target numerical cost. The constraint contains two entities: (1) an operator either 'gt' for greater than , 'lt' for less than or 'eq' for equal to with 10 percent on either side, (2) a target numerical cost. The numerical cost is a number that MUST be defined in the units specified in the ALTO service configuration document obtained from ALTO service discovery. These cost constraints allows a resource constrained ALTO client to filter query results at the ALTO server instead of spending network bandwidth and multiple round trips collecting results and performing client side filtering. If multiple 'constraint' parameters are specified, the ALTO Server assumes they are related to each other with a logical AND.

If the query does not specify the 'type' and 'mode' query string parameters, then the server assumes the type to be 'routingcost' and the mode to be 'numerical'. A Query MUST contain no more than one 'type' parameter, and no more than one 'mode' parameter.

The second form allows an ALTO Client to query for costs from a single Endpoint or PID to all other PIDs:

```
Method         : 'GET'
URI Path       : '/cost/row'
URI QS Params  : 'srcpid=[pidname]'             (multiplicity: 0..*)
                 'srcendp=[endpointname]'       (multiplicity: 0..*)
                 'type=[costtype]'              (multiplicity: 0..1)
                 'mode=[costmode]'              (multiplicity: 0..1)
                 'constraint=[constraint]'      (multiplicity: 0..*)
Headers        : None Required
Body           : MUST be empty
```

Note that in this form, exactly one of 'srcpid' and 'srcendp' query

string parameters MUST be specified.

The third form allows an ALTO Client to query for costs amongst an arbitrary set of sources and destinations:

```
Method         : 'POST'
URI Path       : '/cost/m'
URI QS Params  : 'type=[costtype]'          (multiplicity: 0..1)
                 'mode=[costmode]'          (multiplicity: 0..1)
                 'constraint=[constraint]'  (multiplicity: 0..*)
Headers        : 'Content-Encoding: application/alto'
Body           : See Below
```

In the third form, the Body MUST be an XML document containing the Cost Map Specification XML structure in Appendix B.6.

6.3.4.2. Response

The Response message MUST follow:

```
Status Code    : '200' if all PIDs specified in request are
                       valid, or no PIDs are specified in request.
                 '404' if at least one PID specified in request
                       is not valid.
                 '501' if specified cost type is not supported
                 '501' if constraints not supported but are
                       included
Headers        : 'Content-Encoding: application/alto'
Body           : See Below
```

The Body MUST be an XML document containing the Cost Map XML structure in Appendix B.8.

Note that the ALTO Server is not required to return a 501 code (unsupported query) if an unsupported cost type or cost mode is specified. In such a case, the ALTO Server MAY instead reply with Costs for a default type.

6.3.4.3. Examples of Query and Response

We give two examples. For additional message examples, see Appendix C.3.

```
GET /cost/map HTTP/1.1
Host: alto.example.com

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]
```


```
<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <costmap type="routingcost" mode="ordinal">
    <row srcpid="PID1" size="2">
        <pid name="PID2" cost="5"/>
        <pid name="PID3" cost="10"/>
    </row>
    <row srcpid="PID2" size="2">
        <pid name="PID1" cost="5"/>
        <pid name="PID3" cost="15/>
    </row>
    <row srcpid="PID3" size="2">
        <pid name="PID1" cost="20"/>
        <pid name="PID2" cost="15"/>
    </row>
  </costmap>
</alto>
```


Example Query for Cost Map for All PIDs

```
POST /cost/m?mode=ordinal HTTP/1.1
Host: alto.example.com

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <costmapspec>
    <srcs size="1">
      <endpoint name="ipv4:128.36.22.1"/>
    </srcs>
    <dsts size="3">
      <endpoint name="ipv4:128.30.24.89"/>
      <endpoint name="ipv4:130.132.33.4"/>
      <endpoint name="ipv4:12.32.67.3"/>
    </dsts>
  </costmapspec>
</alto>

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <costmap type="routingcost" mode="ordinal">
    <row srcendp="ipv4:128.36.22.1" size="2">
```

```
          <endpoint name="ipv4:130.132.33.4" cost="1"/>
          <endpoint name="ipv4:128.30.24.89" cost="2"/>
          <endpoint name="ipv4:12.32.67.3" cost="3"/>
        </row>
      </costmap>
    </alto>
```

Example Query for Specific Destinations (Ranking List)

7. Use Cases

The sections below depict typical use cases.

7.1. ALTO Client Embedded in P2P Tracker

Many P2P currently-deployed P2P systems use a Tracker to manage swarms and perform peer selection. P2P trackers may currently use a variety of information to perform peer selection to meet application-specific goals. By acting as an ALTO Client, an P2P tracker can use

ALTO information as an additional information source to enable more network-efficient traffic patterns and improve application performance.

A particular requirement of many P2P trackers is that they must handle a large number of P2P clients. A P2P tracker can obtain and locally store ALTO information (the Network Map and Cost Map) from the ISPs containing the P2P clients, and benefit from the same aggregation of network locations done by ALTO Servers.

```
.---------.   (1) Get Network Map    .---------------.
|         | <----------------------> |               |
|  ALTO   |                          |  P2P Tracker  |
| Server  |   (2) Get Cost Map       | (ALTO Client) |
|         | <----------------------> |               |
`---------'                          `---------------'
                                        ^     |
                          (3) Get Peers |     | (4) Selected Peer
                                        |     v     List
          .---------.                  .-----------.
          | Peer 1  | <--------------- |    P2P    |
```

```
    `---------'                     |  Client   |
        .      (5) Connect to       `-----------'
        .        Selected Peers          /
    .---------.                         /
    | Peer 50 | <------------------
    `---------'
```

Figure 2: ALTO Client Embedded in P2P Tracker

Figure 2 shows an example use case where a P2P tracker is an ALTO Client and applies ALTO information when selecting peers for its P2P clients. The example proceeds as follows:

1. The P2P Tracker requests the Network Map covering all PIDs from the ALTO Server using the Reverse Property Lookup query. The Network Map includes the IP prefixes contained in each PID, allowing the P2P tracker to locally map P2P clients into a PIDs.

2. The P2P Tracker requests the Cost Map amongst all PIDs from the ALTO Server.

3. A P2P Client joins the swarm, and requests a peer list from the P2P Tracker.

4. The P2P Tracker returns a peer list to the P2P client. The returned peer list is computed based on the Network Map and Cost Map returned by the ALTO Server, and possibly other information

sources. Note that it is possible that a tracker may use only the Network Map to implement hierarchical peer selection by preferring peers within the same PID and ISP.

5. The P2P Client connects to the selected peers.

Note that the P2P tracker may provide peer lists to P2P clients distributed across multiple ISPs. In such a case, the P2P tracker may communicate with multiple ALTO Servers.

7.2. ALTO Client Embedded in P2P Client: Numerical Costs

P2P clients may also utilize ALTO information themselves when selecting from available peers. It is important to note that not all

P2P systems use a P2P tracker for peer discovery and selection. Furthermore, even when a P2P tracker is used, the P2P clients may rely on other sources, such as peer exchange and DHTs, to discover peers.

When an P2P Client uses ALTO information, it typically queries only the ALTO Server servicing its own ISP. The my-Internet view provided by its ISP's ALTO Server can include preferences to all potential peers.

```
.---------.   (1) Get Network Map    .---------------.
|         | <----------------------> |               |
|  ALTO   |                          |  P2P Client   |
| Server  |   (2) Get Cost Map       | (ALTO Client) |
|         | <----------------------> |               |    .----------.
`---------'                          `---------------' <- |  P2P     |
          .---------.                 /  |      ^    ^    | Tracker  |
          | Peer 1  | <--------------    |      |     \   `----------'
          `---------'                    |    (3) Gather Peers
              .      (4) Select Peers    |      |       \
              .        and Connect      /   .--------.  .--------.
          .---------.                  /    |  P2P   |  |  DHT   |
          | Peer 50 | <----------------     | Client |  `--------'
          `---------'                       | (PEX)  |
                                            `--------'
```

Figure 3: ALTO Client Embedded in P2P Client

Figure 3 shows an example use case where a P2P Client locally applies ALTO information to select peers. The use case proceeds as follows:

1. The P2P Client requests the Network Map covering all PIDs from the ALTO Server servicing its own ISP.

2. The P2P Client requests the Cost Map amongst all PIDs from the ALTO Server. The Cost Map by default specifies numerical costs.

3. The P2P Client discovers peers from sources such as Peer Exchange (PEX) from other P2P Clients, Distributed Hash Tables (DHT), and P2P Trackers.

4. The P2P Client uses ALTO information as part of the algorithm for selecting new peers, and connects to the selected peers.

7.3. ALTO Client Embedded in P2P Client: Ranking

It is also possible for a P2P Client to offload the selection and ranking process to an ALTO Server. In this use case, the ALTO Client gathers a list of known peers in the swarm, and asks the ALTO Server to rank them.

As in the use case using numerical costs, the P2P Client typically only queries the ALTO Server servicing its own ISP.

```
   .---------.                          .---------------.
   |         |                          |               |
   |  ALTO   |   (2) Get Path Ranking   |  P2P Client   |
   | Server  | <----------------------> | (ALTO Client) |
   |         |                          |               |    .---------.
   `---------'                          `---------------' <- |  P2P    |
             .---------.                 /  |      ^    ^    | Tracker |
             | Peer 1  | <--------------    |      |     \   `---------'
             `---------'                    |    (1) Gather Peers
                 .      (3) Connect to      |      |       \
                 .        Selected Peers   /   .--------.  .--------.
             .---------.                  /    |  P2P   |  |  DHT   |
             | Peer 50 | <----------------     | Client |  `--------'
             `---------'                       | (PEX)  |
                                               `--------'
```

Figure 4: ALTO Client Embedded in P2P Client: Ranking

Figure 4 shows an example of this scenario. The use case proceeds as follows:

1. The P2P Client discovers peers from sources such as Peer Exchange (PEX) from other P2P Clients, Distributed Hash Tables (DHT), and P2P Trackers.

2. The P2P Client queries its ALTO Server, including discovered peers as the set of Destination Network Locations, and indicates the 'ordinal' Cost Mode. The returned Cost Map indicates the

ranking of the candidate peers.

3. The P2P Client connects to the peers in the order specified in the ranking.

8. Discussions

8.1. Discovery

The particular mechanism by which an ALTO Client discovers its ALTO Server is an important component to the ALTO architecture and numerous techniques have been discussed [13] [14]. However, the discovery mechanism is out of scope for this document.

Some ISPs have proposed the possibility of delegation, in which an ISP provides information for customer networks which do not wish to run Portal Servers themselves. A consideration for delegation is that customer networks may wish to explicitly configure such delegation.

8.2. Network Address Translation Considerations

At this day and age of NAT v4<->v4, v4<->v6 [15], and possibly v6<->v6[16], a protocol should strive to be NAT friendly and minimize carrying IP addresses in the payload, or provide a mode of operation where the source IP address provide the information necessary to the server.

The protocol specified in this document provides a mode of operation (the GetCostMap-Source interface) where the source NL-ID is computed by the ALTO Server (via the Endpoint Property Lookup interface) from the source IP address found in the ALTO Client query packets. This is similar to how some P2P Trackers (e.g., BitTorrent Trackers - see "Tracker HTTP/HTTPS Protocol" in [17]).

The ALTO client SHOULD use the Session Traversal Utilities for NAT (STUN) [4] to determine a public IP address to use as a source NL-ID. If using this method, the host MUST the "Binding Request" message and the resulting "XOR-MAPPED-ADDRESS" parameter that is returned in the response. Using STUN requires cooperation from a publicly accessible STUN server. Thus, the ALTO client also requires configuration information that identifies the STUN server, or a domain name that can be used for STUN server discovery. To be selected for this purpose, the STUN server needs to provide the public reflexive transport address of the host.

8.3. Mapping IPs to ASNs

It may be desired for the ALTO Protocol to provide ALTO information including ASNs. Thus, ALTO Clients may need to identify the ASN for a Resource Provider to determine the cost to that Resource Provider.

Applications can already map IPs to ASNs using information from a BGP Looking Glass. To do so, they must download a file of about 1.5MB when compressed (as of October 2008, with all information not needed for IP to ASN mapping removed) and periodically (perhaps monthly) refresh it.

Alternatively, Reverse Property Lookup query defined in this document could be extended to map ASNs into a set of IP prefixes. The mappings provided by the ISP would be both smaller and more authoritative.

For simplicity of implementation, it's highly desirable that clients only have to implement exactly one mechanism of mapping IPs to ASNs.

8.4. Endpoint and Path Properties

An ALTO Server could make available many properties about Endpoints beyond their network location or grouping. For example, connection type, geographical location, and others may be useful to applications. The current draft focuses on network location and grouping, but the protocol may be extended to handle other Endpoint properties.

8.5. P2P Peer Selection

This section discusses possible approaches to peer selection using ALTO information (Network Location Identifiers and associated Costs) from an ALTO Server. Specifically, the application must select which peers to use based on this and other sources of information. With this in mind, the usage of ALTO Costs is intentionally flexible, because:

> Different applications may use the information differently. For example, an application that connects to just one address may have a different algorithm for selecting it than an application that connects to many.

Though initial experiments have been conducted [18], more investigation is needed to identify other methods.

In addition, the application might account for robustness, perhaps using randomized exploration to determine if it performs better

without ALTO information.

8.5.1. Client-based Peer Selection

One possibility is for peer selection using ALTO costs to be done entirely by a P2P client. The following are some techniques have been proposed and/or used:

- o Prefer network locations with lower ordinal rankings (i.e., higher priority) [19] [8].

- o Optimistically unchoking low-cost peers with higher probability [8].

8.5.2. Server-based Peer Selection

Another possibility is for ALTO costs to be used by an Application Tracker (e.g., BitTorrent Tracker) when returning peer lists. The following are techniques that have been proposed and/or used:

- o Using bandwidth matching (e.g., at an Application Tracker) and choosing solution (within bound of optimal) with minimal network cost [18].

9. IANA Considerations

This document request the registration of a new media type: "application/alto"

10. Security Considerations

10.1. ISPs

ISPs must be cognizant of the network topology and provisioning

information provided through ALTO Interfaces. ISPs should evaluate how much information is revealed and the associated risks. In particular, providing overly fine-grained information may make it easier for attackers to infer network topology. On the other hand, revealing overly coarse-grained information may not provide benefits to network efficiency or performance improvements to ALTO Clients.

10.2. ALTO Clients

Applications using the information must be cognizant of the possibility that the information is malformed or incorrect. Even when it is correct, its use might harm the performance. When an

application concludes that it would get better performance disregarding the ALTO information, the decision to discontinue the use of ALTO information is likely best left to the user.

ALTO Clients should also be cognizant of revealing Network Location Identifiers (IP addresses or fine-grained PIDs) to the ALTO Server, as doing so may allow the ALTO Server to infer communication patterns. One possibility is for the ALTO Client to only rely on Network Map for PIDs and Cost Map amongst PIDs to avoid passing IP addresses of their peers to the ALTO Server.

The use of SSL/TLS can make it easier for clients to verify the origin of ALTO information.

10.3. ALTO Information

An ALTO Server may optionally use authentication and encryption to protect ALTO information. Authentication and encryption may be provided using HTTP Basic or Digest Authentication and/or SSL/TLS.

10.4. ALTO Information Redistribution

It is possible for applications to redistribute ALTO information to improve scalability. Even with such a distribution scheme, ALTO Clients obtaining ALTO information must be able to validate the received ALTO information to ensure that it was actually generated by the correct ALTO Server. Further, to prevent the ALTO Server from being a target of attack, the verification scheme must not require ALTO Clients to contact the ALTO Server.

To fulfill these requirements, ALTO Information meant to be redistributable contains a digital signature which includes a hash of the ALTO information encrypted by the ALTO Server's private key. The corresponding public key should either be part of the ALTO information itself, or it could be included in the interface descriptor. The public key SHOULD include the hostname of the ALTO Server and it SHOULD be signed by a trusted authority.

11. References

11.1. Normative References

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

[2] Berners-Lee, T., Fielding, R., and H. Nielsen, "Hypertext Transfer Protocol -- HTTP/1.0", RFC 1945, May 1996.

[3] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[4] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for (NAT) (STUN)", draft-ietf-behave-rfc3489bis-18 (work in progress), July 2008.

11.2. Informative References

[5] Kiesel, S., Popkin, L., Previdi, S., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", draft-kiesel-alto-reqs-01 (work in progress), November 2008.

[6] Alimi, R., Pasko, D., Popkin, L., Wang, Y., and Y. Yang, "P4P: Provider Portal for P2P Applications", draft-p4p-framework-00 (work in progress), November 2008.

[7] Wang, Y., Alimi, R., Pasko, D., Popkin, L., and Y. Yang, "P4P Protocol Specification", draft-wang-alto-p4p-specification-00 (work in progress), March 2009.

[8] Shalunov, S., Penno, R., and R. Woundy, "ALTO Information Export Service", draft-shalunov-alto-infoexport-00 (work in progress), October 2008.

[9] Das, S. and V. Narayanan, "A Client to Service Query Response Protocol for ALTO", draft-saumitra-alto-queryresponse-00 (work in progress), March 2009.

[10] Das, S., Narayanan, V., and L. Dondeti, "ALTO: A Multi Dimensional Peer Selection Problem", draft-saumitra-alto-multi-ps-00 (work in progress), October 2008.

[11] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", draft-marocco-alto-problem-statement-04 (work in progress), February 2009.

[12] Yang, Y., Popkin, L., Penno, R., and S. Shalunov, "An Architecture of ALTO for P2P Applications", draft-yang-alto-architecture-00 (work in progress), March 2009.

[13] Garcia, G., Tomsu, M., and Y. Wang, "ALTO Discovery Protocols", draft-wang-alto-discovery-00 (work in progress), March 2009.

[14] Song, H., Even, R., Pascual, V., and Y. Zhang, "Application-

Layer Traffic Optimization (ALTO): Discover ALTO Servers", draft-song-alto-server-discovery-00 (work in progress), March 2009.

[15] Baker, F., Li, X., and C. Bao, "Framework for IPv4/IPv6 Translation", draft-baker-behave-v4v6-framework-02 (work in progress), February 2009.

[16] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Address Translation (NAT66)", draft-mrw-behave-nat66-02 (work in progress), March 2009.

[17] "Bittorrent Protocol Specification v1.0", http://wiki.theory.org/BitTorrentSpecification, 2009.

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

[19] Akonjang, O., Feldmann, A., Previdi, S., Davie, B., and D. Saucez, "The PROXIDOR Service", draft-akonjang-alto-proxidor-00 (work in progress), March 2009.

Appendix A. Data Types

A.1. Endpoint Name

TBD.

A.2. PID Name

TBD.

A.3. Property Name

TBD.

A.4. IP Prefix

TBD.

A.5. Cost Type

TBD.

A.6. Cost Mode

TBD.

A.7. Constraint

TBD.

Appendix B. XML Encoding

B.1. Server Configuration

The Response contains a 'configuration' XML element which contains the configuration information for an ALTO. service. The 'configuration' element MUST have the following attributes:

o name of the ALTO service

The 'configuration' element MAY contain the following child elements:

o specifies in its 'uri' attribute, the Base URI at which the ALTO Server can be reached. An ALTO Client uses this URI (e.g., 'http://alto.example.com:6671/') as a prefix placed before URI Paths when querying an ALTO Server. More than one 'alto-server' element may be present for load balancing, and an ALTO Client can choose any one at random.

o specifies a cost metric supported by the ALTO Server. It MUST have a 'type' attribute indicating the name of the metric, and MUST have a 'units' attribute indicating the measurement units. If the metric does not have any units, then the units attribute must have the value 'none'. unit. If the no 'cost' element is present, then the ALTO Server is assumed to support the default 'routingcost' Cost metric. Multiple 'cost' elements MAY be included, but a single Cost Type MUST NOT appear more than once.

o specifies whether the ALTO Server supports Cost constraints in the Path Cost Lookup Query Section 6.3.4. This element MUST contain a 'value' attribute with value either 'true' or 'false'. The 'constraint-support' element MUST NOT appear more than once. If the 'constraint-support' element is not present, the ALTO Client MUST assume that the ALTO Server does not support Cost constraints.

B.2. Endpoint

An Endpoint is represented by the XML element 'endpoint'. The following attributes are REQUIRED:

name: Indicates the name of the Endpoint. The value of this attribute MUST be formatted according to Appendix A.1.

The 'endpoint' element MAY contain additional attributes indicating endpoint properties and their values. In this case, the attribute name is the property name, and the attribute value is the value of the property. Note that 'name' is not a valid property name.

B.3. Endpoint List

A list of Endpoints is represented by the XML element 'endpoints'. The following attributes are REQUIRED:

size: Specifies the number of endpoints contained in the list as a non-negative integer.

The 'endpoints' element MAY contain child elements. The following elements are allowed:

element: Specifies a single endpoint in the list. The number of 'endpoint' elements MUST equal the value of the 'size' attribute for the containing 'endpoints' element.

B.4. PID

TBD.

B.5. PID List

TBD.

B.6. Cost Map Specification

TBD.

B.7. Cost Row

TBD.

B.8. Cost Map

TBD.

Appendix C. Additional Protocol Message Examples

C.1. Endpoint Property Lookup

```
POST /endpoint/m?prop=pid HTTP/1.1
Host: alto.example.com
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <endpoints size="2">
    <endpoint name="ipv4:128.36.1.34"/>
    <endpoint name="ipv4:130.132.1.100"/>
  </endpoints>
</alto>

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <endpoints size="2">
    <endpoint name="ipv4:128.36.1.34" pid="PID1"/>
    <endpoint name="ipv4:130.132.1.100" pid="PID2"/>
  </endpoints>
</alto>
```

Example Query for Multiple Endpoints

C.2. Reverse Property Lookup

```
GET /prop/pid/ HTTP/1.1
Host: alto.example.com

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]
```


```
<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <pids size="3">
    <pid name="PID1" size="3">
      <cidr4 name="128.36.1.0/24"/>
      <cidr4 name="132.130.1.0/24"/>
      <cidr4 name="132.130.2.0/24"/>
    </pid>
    <pid name="PID2" size="1">
      <cidr4 name="130.132.3.0/24"/>
    </pid>
    <pid name="PID3" size="1">
      <cidr4 name="0.0.0.0/0"/>
    </pid>
  </pids>
</alto>
```


Example Query for All PIDs

```
POST /prop/pid/m HTTP/1.1
Host: alto.example.com
Content-Length: [...]

<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <pids size="2">
    <pid name="PID1"/>
    <pid name="PID2"/>
  </pids>
</alto>

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <pids size="2">
    <pid name="PID1" size="3">
      <cidr4 name="128.36.1.0/24"/>
      <cidr4 name="132.130.1.0/24"/>
      <cidr4 name="132.130.2.0/24"/>
    </pid>
    <pid name="PID2" size="1">
      <cidr4 name="130.132.3.0/24"/>
    </pid>
  </pids>
</alto>
```

Example Query for Specific PIDs

C.3. Path Cost Lookup

```
GET /cost/row?srcendp=ipv4:128.36.22.1 HTTP/1.1
Host: alto.example.com

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: application/alto
Content-Length: [...]

<?xml version="1.0" encoding="UTF-8"?>
<?oxygen RNGSchema="config-schema.rnc" type="compact"?>
<alto xmlns="urn:ietf:params:xml:ns:p2p:alto">
  <costmap type="routingcost" mode="ordinal">
    <row srcendp="ipv4:128.36.22.1" size="2">
        <pid name="PID1" cost="1"/>
        <pid name="PID2" cost="5"/>
        <pid name="PID3" cost="10"/>
    </row>
  </costmap>
</alto>
```

Example Query for Cost Map from a Single Endpoint

Appendix D. Contributors

The people listed here should be viewed as co-authors of the document. Due to the limit of 5 authors per draft the co-authors were moved to the contributors section at this point.

Obi Akonjang

DT Labs/TU Berlin/

EMail: obi@net.t-labs.tu-berlin.de

Richard Alimi

Yale University

EMail: richard.alimi@yale.edu

Saumitra M. Das

Qualcomm Inc.

EMail: saumitra@qualcomm.com

Syon Ding

China Telecom

EMail: syding@chinatelecom.com

Doug Pasko

Verizon

EMail: pasko@verizon.com

Laird Popkin

Pando Networks

EMail: laird@pando.com

Stefano Previdi

Cisco

EMail: sprevidi@cisco.com

Satish Raghunath

Juniper Networks

satishr@juniper.net

Stanislav Shalunov

BitTorrent

EMail: shalunov@bittorrent.com

Albert Tian

Ericsson/Redback

EMail: alberttian@gmail.com

Yu-Shun Wang

Microsoft Corp.

yu-shun.wang@microsoft.com

Richard Woundy

Comcast

Richard_Woundy@cable.comcast.com

David Zhang

PPLive

davidzhang@pplive.com

Yunfei Zhang

China Mobile

zhangyunfei@chinamobile.com

Appendix E. Acknowledgements

We would like to thank the following additional people who were involved in the projects that contributed to this merged document:

Alex Gerber (AT&T), Chris Griffiths (Comcast), Ramit Hora (Pando Networks), Arvind Krishnamurthy (University of Washington), Marty Lafferty (DCIA), Erran Li (Bell Labs), Jin Li (Microsoft), Y. Grace Liu (IBM Watson), Jason Livingood (Comcast), Michael Merritt (AT&T), Ingmar Poese (DT Labs/TU Berlin), James Royalty (Pando Networks), Damien Saucez (UCL) Thomas Scholl (AT&T), Emilio Sepulveda (Telefonica), Avi Silberschatz (Yale University), Hassan Sipra (Bell Canada), Georgios Smaragdakis (DT Labs/TU Berlin), Haibin Song (Huawei), Oliver Spatscheck (AT&T), See-Mong Tang (Microsoft), Jia Wang (AT&T), Hao Wang (Yale University), Ye Wang (Yale University), Haiyong Xie (Yale University).

Authors' Addresses

Reinaldo Penno (editor)
Juniper Networks
1194 N Mathilda Avenue
Sunnyvale,   CA
USA

Email: rpenno@juniper.net

Y. Richard Yang (editor)
Yale University

Email: yry@cs.yale.edu