ALTO
Internet-Draft
Intended status: Informational
Expires: January 12, 2012

M. Stiemerling
NEC Europe Ltd.
S. Kiesel
University of Stuttgart

July 11, 2011

ALTO Deployment Considerations
draft-ietf-alto-deployments-02

Abstract

Many Internet applications are used to access resources, such as pieces of information or server processes, which are available in several equivalent replicas on different hosts. This includes, but is not limited to, peer-to-peer file sharing applications. The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to these applications, which have to select one or several hosts from a set of candidates, that are able to provide a desired resource. The protocol is under specification in the ALTO working group. This memo discusses deployment related issues of ALTO for peer-to-peer and CDNs, some preliminary security considerations, and also initial guidance for application designers using ALTO.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on January 12, 2012.

Table of Contents

1. Introduction

Many Internet applications are used to access resources, such as pieces of information or server processes, which are available in several equivalent replicas on different hosts. This includes, but is not limited to, peer-to-peer file sharing applications and Content Delivery Networks (CDNs). The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications, which have to select one or several hosts from a set of candidates, that are able to provide a desired resource. The basic ideas of ALTO are described in the problem space of ALTO is described in [RFC5693] and the set of requirements is discussed in [I-D.ietf-alto-reqs].

However, there are no considerations about what operational issues are to be expected once ALTO will be deployed. This includes, but is not limited to, location of the ALTO server, imposed load to the ALTO server, or from whom the queries are performed.

Comments and discussions about this memo should be directed to the ALTO working group: alto@ietf.org.

2. General Considerations

The ALTO protocol is a client/server protocol, operating between a number of ALTO clients and an ALTO server, as sketched in Figure 1. The ALTO working groups defines the ALTO protocol [I-D.ietf-alto-protocol].

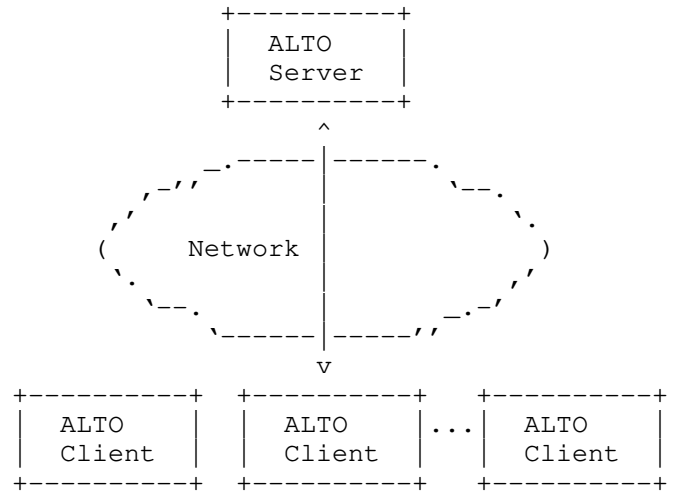


Figure 1: Network Overview of ALTO Protocol

2.1. General Placement of ALTO

The ALTO server and ALTO clients can be situated at various entities in a network deployment. The first differentiation is whether the ALTO client is located on the actual host that runs the application, as shown in Figure 2, (e.g., peer-to-peer filesharing application) or if the ALTO client is located on resource directory, as shown in Figure 3 (e.g., a tracker in peer-to-peer filesharing).

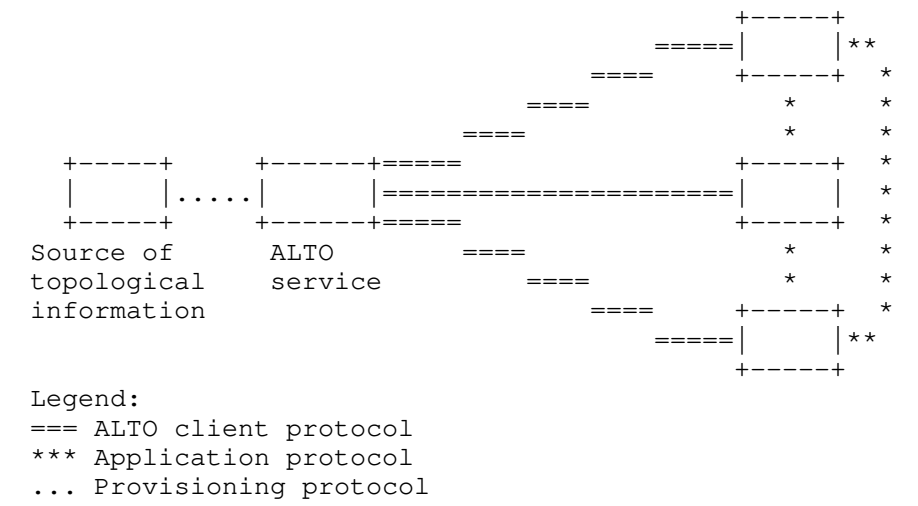


Figure 2: Overview of protocol interaction between ALTO elements,scenario without tracker

Figure 2 shows the operational model for applications that do not use a tracker, such as, edonky, or in if the tracker should be the querying party. This use case also holds true for CDNs. The ALTO server can also be queried by CDNs to get a guidance about where the a particular client accessing data in the CDN is exactly located in the ISP's network.

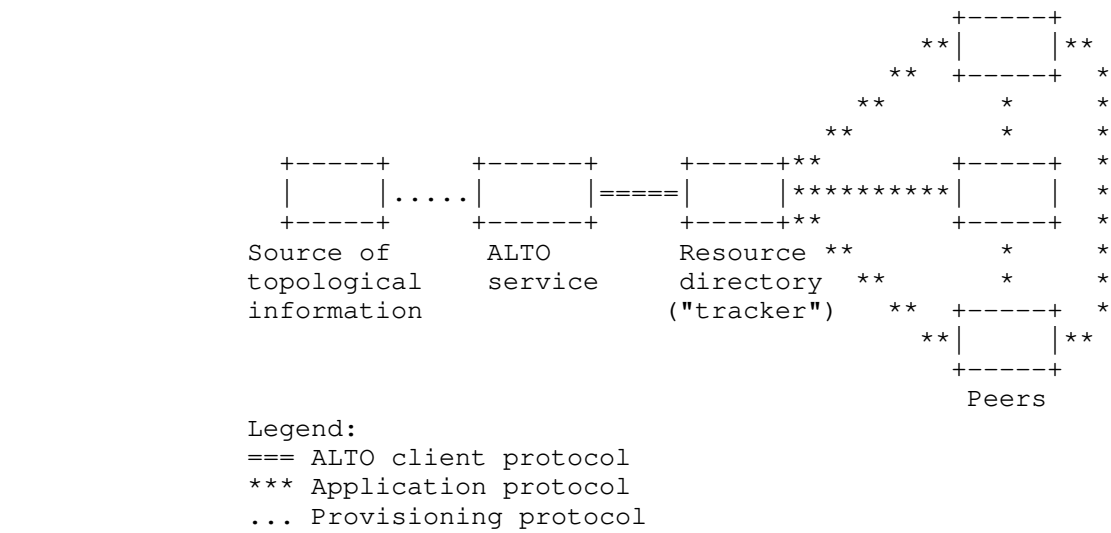


Figure 3: Overview of protocol interaction between ALTO elements, scenario with tracker

However, Figure 3 does not denote where the ALTO elements are actually located, i.e., if the tracker and the ALTO server are in the same ISP's domain, or if the tracker and the ALTO server are managed/ owned/located in different domains. The latter is the typical use case, e.g., taking Pirate Bay as example that serves Bittorrent peers world-wide.

2.2. Relationship between ALTO and Applications

ALTO is intended to be used by a wide-range of applications. However, any application using ALTO must also work if no ALTO servers can be found or if no responses to ALTO queries are received, e.g., due to connectivity problems or overload situation (see also [I-D.ietf-alto-reqs]). (Editor's note: better text needed here!)

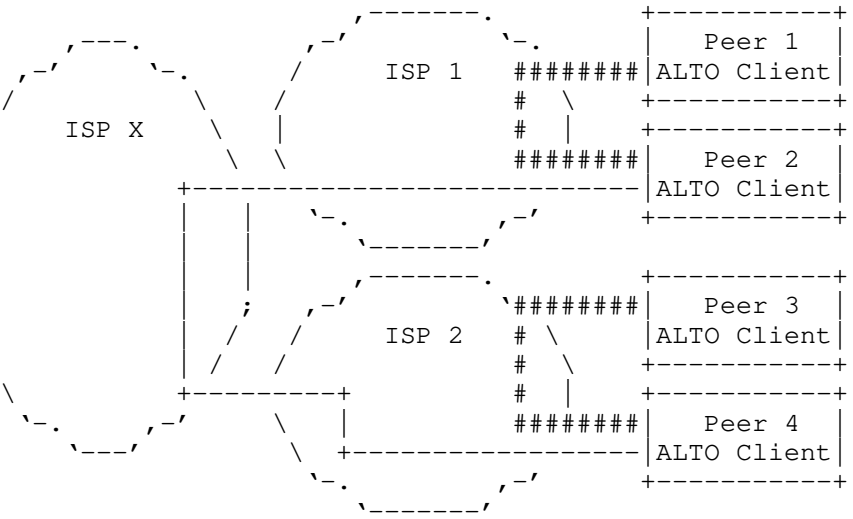
2.3. Provided Guidance

ALTO gives guidance to applications on what IP addresses or IP prefixes, and such which hosts are to be preferred according to the operator of the ALTO server. The general assumption of the ALTO WG is that a network operator would always express to prefer hosts in its own network while hosts located outside its own network are to be avoided (are undesired to be considered by the applications). This might be applicable in some cases but may not be applicable in the general case. The ALTO protocol gives only the means to let the ALTO server operator to express is preference, whatever this preference

is. This section explores this space.

2.3.1. Keeping Traffic Local in Network

ALTO guidance can be used to let applications prefer other peers within the same network operator's network instead of randomly connecting to other peers which are located in another operator's network. Figure 4 shows such a scenario where peers prefer peers in the same network (e.g., Peer 1 and Peer 2 in ISP1 and Peer 3 and Peer 4 in ISP2).



Legend:
preferred "connections"
--- non-preferred "connections"

Figure 4: ALTO Traffic Network Localization

TBD: Describes limits of this approach (e.g., traffic localization guidance is of less use if the peers cannot upload); describe how maps would look like.

2.3.2. Off-Loading Traffic from Network

Another scenario where the use of ALTO can be beneficial is in mobile broadband networks, e.g., CDMA200 or UMTS, but where the network operator may have the desire to guide peers in its own network to use peers in remote networks. One reason can be that the wireless network is not made for the load cause by, e.g., peer-to-peer

applications, and the operator has the need that peers fetch their data from remote peers in other parts of the Internet.

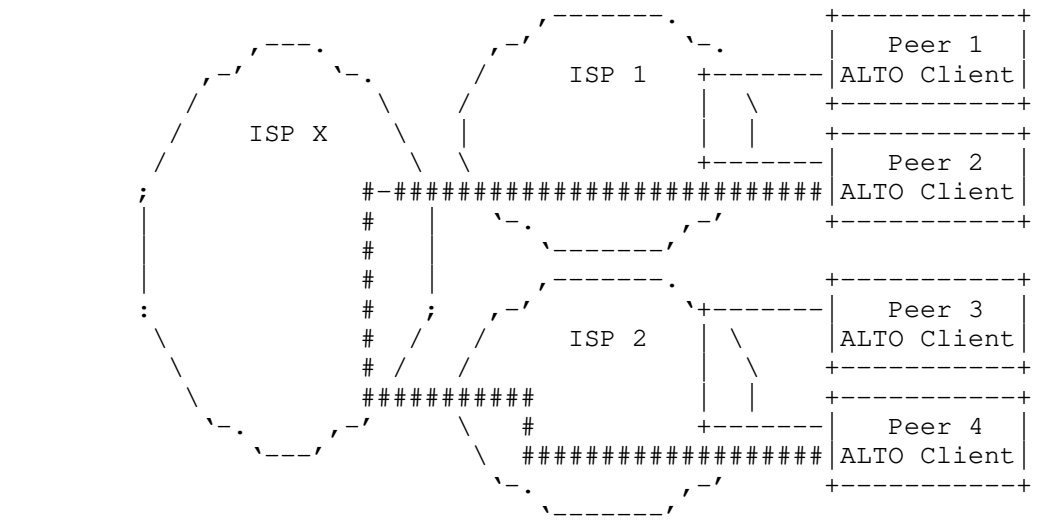


Figure 5: ALTO Traffic Network De-Localization

Figure 5 shows the result of such a guidance process where Peer 2 prefers a connection with Peer4 instead of Peer 1, as shown in Figure 4.

TBD: Limits of this approach in general and with respect to p2p. describe how maps would look like.

2.3.3. Intra-Network Localization/Bottleneck Off-Loading

The above sections described the results of the ALTO guidance on an inter-network level. However, ALTO can also be used to guide peers on which internal peers are to be preferred. For instance, to guide Peers on a remote network side to prefer to connect to each other, instead of crossing a bottleneck link, a backhaul link to connect the side to the network core. Figure 6 shows such a scenario where Peer 1 and Peer 2 are located in Net 2 of ISP1 and connect via a low capacity link to the core (Net 1) of the same ISP1. Peer1 and Peer 2 would both exchange their data with remote peers, probably clogging the bottleneck link.

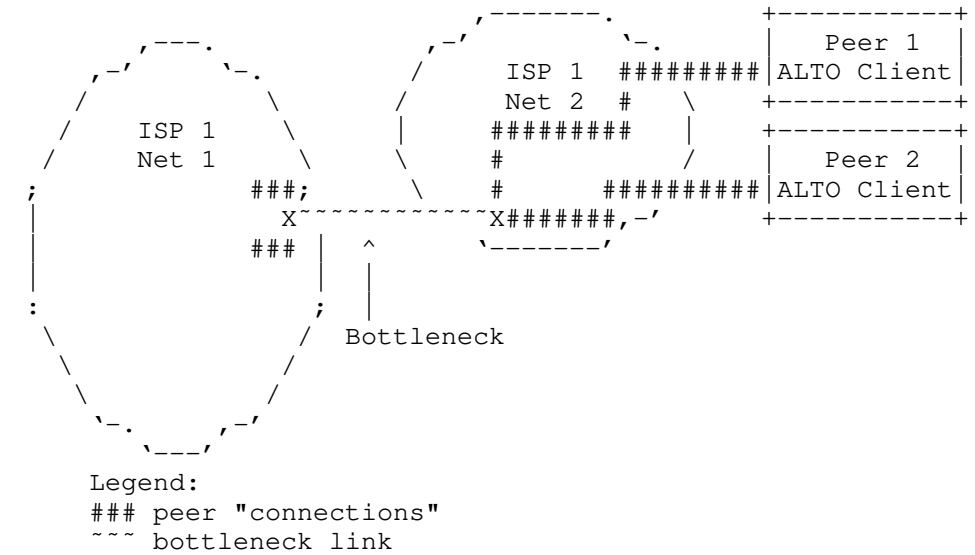


Figure 6: Without Intra-Network ALTO Traffic Localization

The operator can guide the peers in such a situation to try first local peers in the same network islands, avoiding or at least lowering the effect on the bottleneck link, as shown in Figure 7.

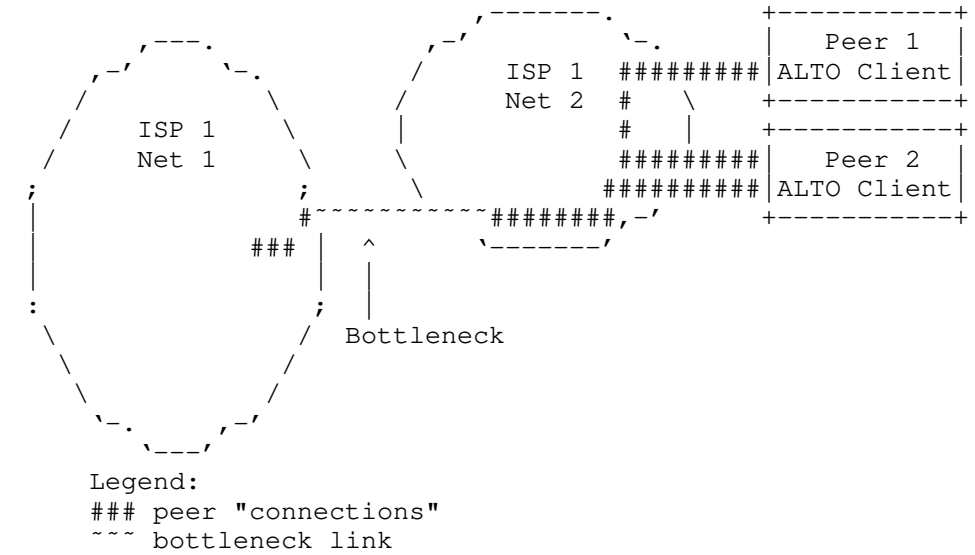


Figure 7: With Intra-Network ALTO Traffic Localization

TBD: describe how maps would look like.

2.4. Provisiong ALTO Maps

This section will describe how ALTO maps in the protocol can be populated before using them.

3. Deployment Considerations by ISPs

The Internet is a large network constituted of multiple networks worldwide. Numerous of these networks are built by telecom operators or network operators (named ISP in this memo), and these networks provide network connectivity, such as cable networks, 3G and so on. As well as some of networks are built by universities or big organizations themselves, and these networks are used to provide connectivity for research and work. The essence of Internet is its connectivity and sharing capability. However, ISPs emphasize network's manageability and controllability, because ISPs provide public network access service for most person and families, they need to manage, to control and to audit the traffic. Thus, it's important for ISPs to understand the requirement of optimizing traffic, and how to deploy ALTO service in these manageability and controllability networks.

3.1. Requirement for Traffic Optimization by ISPs

All networks of ISPs are connected to each other through peering points. From view of business mode, the inter-network settlement is needed in traffic exchanging between these ISP's networks. The current settlement can be costly. So to save these cost, the simple and basic method is to decrease the traffic exchange across the peering points and keep the traffic in own network area.

For some large ISPs, their whole network is layered. The upper layer network includes one or several backbone networks, and the lower layer network includes multiple access networks. These access networks are connected to backbone networks, and the exchange traffic with others through backbone network. In this kind of layered network, the bandwidth of backbone network is important and may be scarce. Traffic should be limited to the access networks, so to decrease the usage of backbone as far as possible.

Compared to fixed networks, mobile networks have some special characters, including small link bandwidth, high cost, limited radio frequency resource, and terminal battery. In mobile network, the usage of wireless link should be decreased as far as possible and be high-efficient. For example, in the case of a P2P service, the clients in the fixed network should decrease the data transport from the clients in the mobile networks, as well as the clients in the mobile networks should prefer the data transmission from the clients in the fixed networks.

3.2. Considerations for ISPs

3.2.1. Very small ISPs with simple Network Structure

For very small ISPs, the traffic optimizing problem they focus is that how to decrease the traffic exchanging with other ISPs, because of high settlement costs. To use the ALTO service to optimize traffic, small ISPs can define two optimization areas: one is their own network; the other is all outer networks connected with their network. The cost map can be defined like this: the cost of link between clients of inner ISP's networks is lower than from clients of outer ISP's networks to clients of inner ISP's networks. So the client of this ISP will prefer to require data from the clients in the same ISP with high priority.

One example is given as below in Figure 8. ISP A is one small ISP, only having one access network. In ALTO service deploying, we can define ISP A to be one optimization area, named as PID1, and define other networks to be the other optimization area, named as PID2. C1 is denoted as the link cost in inner ISP A. C2 is denoted as the link cost from PID2 to PID1. We define the cost map as:

C1<C2

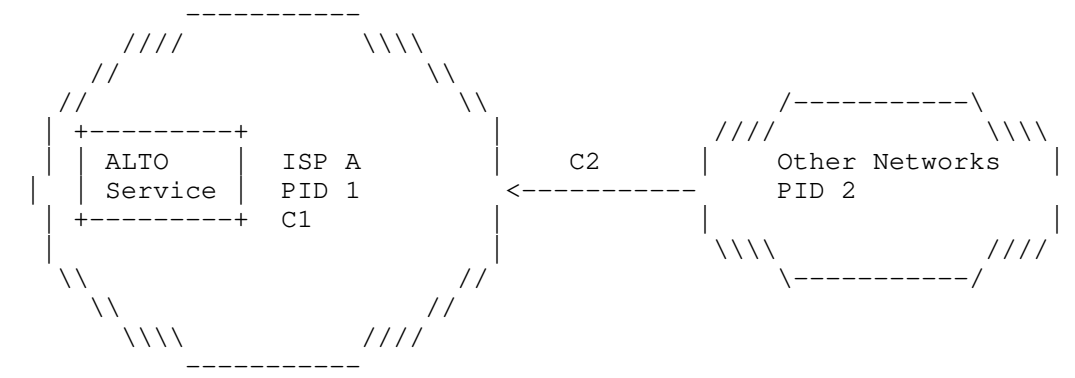


Figure 8: ALTO deployment in small ISPs

3.2.2. Large ISPs with layered fixed Network Structure

For large ISPs with layered fixed network structure, the traffic optimizing problems they focus will include that: using backbone network by high-efficiency, adjusting traffic balance in different access networks according to traffic conditions and management policies, and considering settlement cost with other ISPs. So in

ALTO service deploying to this kind of large ISP, first the optimization area can be defined according to real network condition. For example, each access network can be defined to be one optimization area. Then cost can be defined according to the optimizing requirement by ISPs. There is one example described below and also shown in Figure 9.

In this example, ISP A has one backbone network and three access networks, named as AN A, AN B, and AN C. A P2P application is used in this example. For the traffic optimization, the first requirement is to decrease the P2P traffic of backbone network in inner ISP A; and the second requirement is to decrease the P2P traffic to outer ISPs. Always, the second requirement is prior to the first one. Also, we assume that the settlement rate with ISP B is lower than with other ISPs. Then ISP A can deploy ALTO service to meet the need of traffic optimization. We will give the detail example of ALTO service definition and configuration according to requirements above.

In inner network of ISP A, we can define each access network to be one optimization area, and assign one PID to every access network, such as PID1, PID2, and PID 3. Because of different settlement with different outer ISPs, we define ISP B to be one optimization area, and assign PID 4 to it, as well as define all other networks to be one optimization area and PID 5.

We assign cost names (C1, C2, C3, C4, C5, C6, C7) as the figure below. C1 is denoted as the link cost in inner AN A, the same as C2 and C3. C4 is denoted as the link cost from PID 1 to PID 2, the same as C5. C6 is denoted as the link cost from the ISP B to ISP A. C7 is denoted as the link cost from other networks to ISP A.

According to discussion of the first requirement and the second requirement above, the relationship of these costs will be defined as: (C1, C2, C3) < (C4, C5) < (C6) < (C7)

This is one very simple example above, in which we do not consider the different link type of access network. In deploying ALTO service in real network, we must consider more real network conditions and requirements. One real example is described in greater detail in [I-D.lee-alto-chinatelecom-trial].

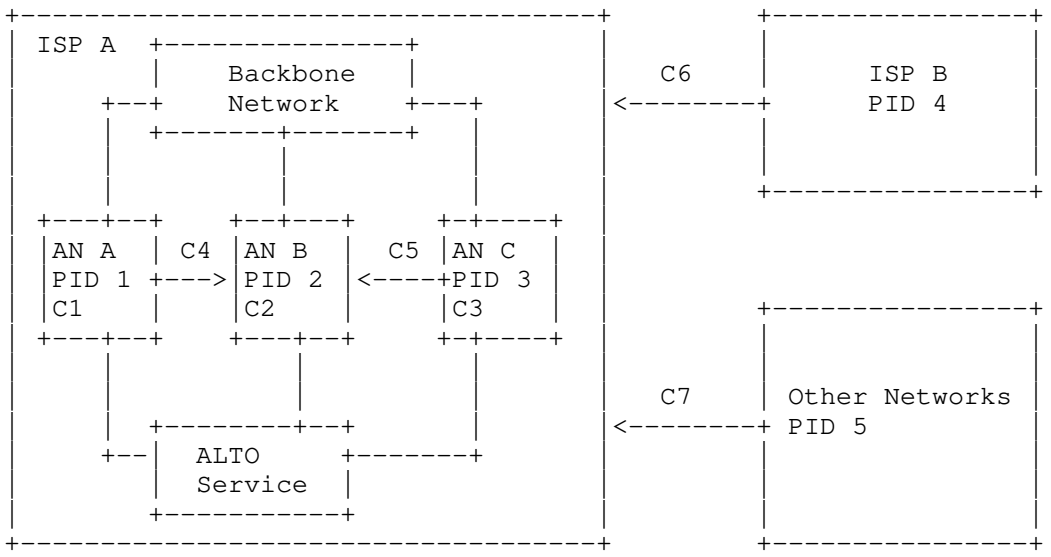


Figure 9: ALTO deployment in large ISPs with layered fixed network structures

3.2.3. ISPs with Mobile Network

For ISPs with mobile network and fixed network, the traffic optimizing problems they focus will be optimizing the mobile traffic, except problems on last hop section. Wireless radio frequency resource is scarce and costly in mobile network. The requirement of traffic optimization in mobile network is mainly decreasing the usage of radio resource. The ALTO service can be deployed to meet these needs.

For example in one ISP A as below in Figure 10, there is one mobile network is connected to backbone network. In this kind of network structure, mobile network can be defined as one optimization area, and assigned PID 1. We also define other PID and cost as figure below.

To decrease the usage of wireless link, the relationship of these costs will be defined to:

From view of mobile network:(C4 < C1). This means that, the clients in mobile network requiring data resource from clients of the other access networks is prior to clients of mobile network. This policy can decrease the usage of wireless link and power consumption in terminal.

From view of AN A:(C2 < C6, C5 = maximum cost). This means that, to other optimization area, requiring data from mobile network should be avoided.

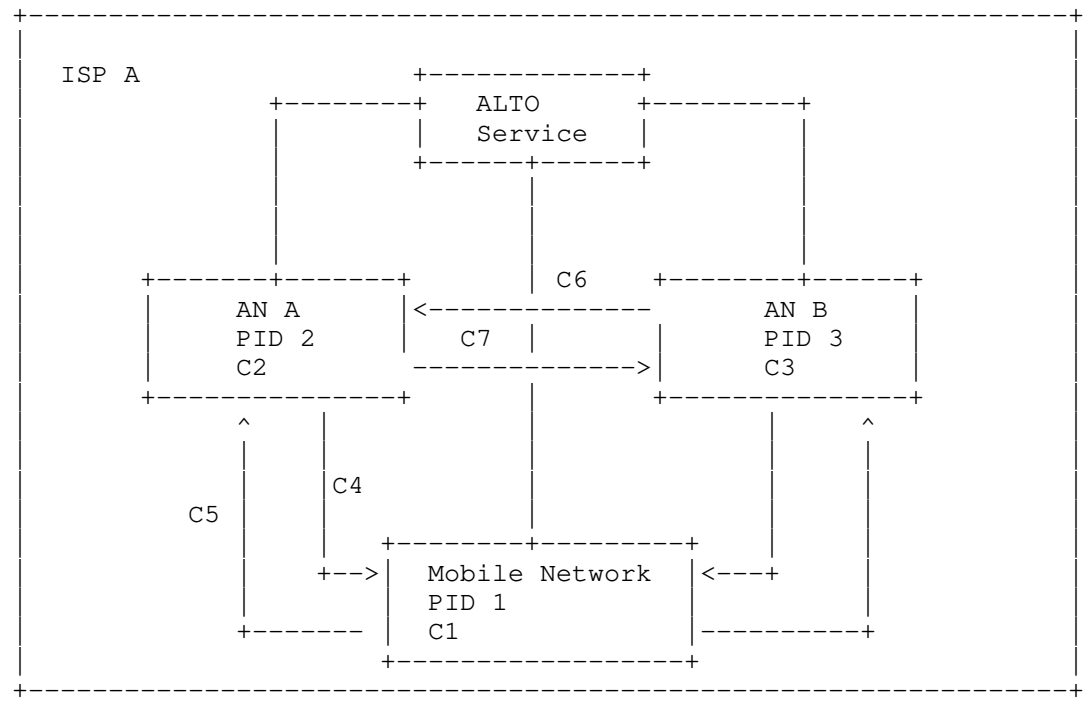


Figure 10: ALTO deployment in ISPs with mobile network

4. Using ALTO for P2P

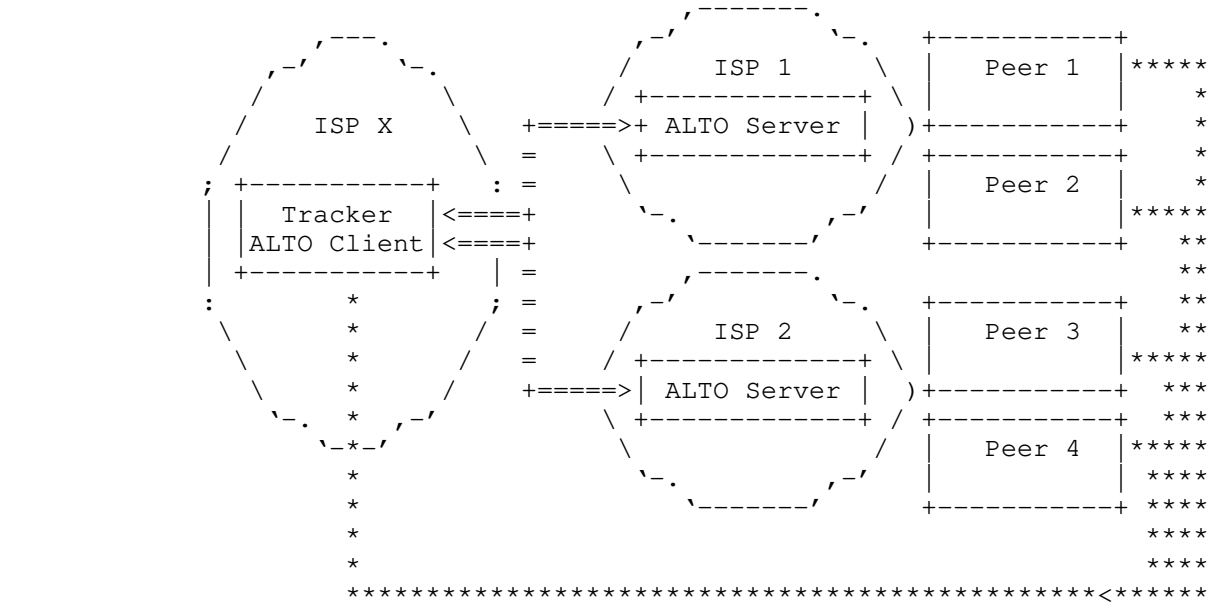


Figure 11: Global tracker accessing ALTO server at various ISPs

Figure 11 depicts a tracker-based system, where the tracker embeds the ALTO client. The tracker itself is hosted and operated by an entity different than the ISP hosting and operating the ALTO server. Initially, the tracker has to look-up the ALTO server in charge for each peer where it receives a ALTO query for. Therefore, the ALTO server has to discover the handling ALTO server, as described in [I-D.kiesel-alto-3pdisc]. However, the peers do not have any way to query the server themselves. This setting allows to give the peers a better selection of candidate peers for their operation at an initial time, but does not consider peers learned through direct peer-to-peer knowledge exchange, AKA peer exchange in various peer-to-peer protocols.

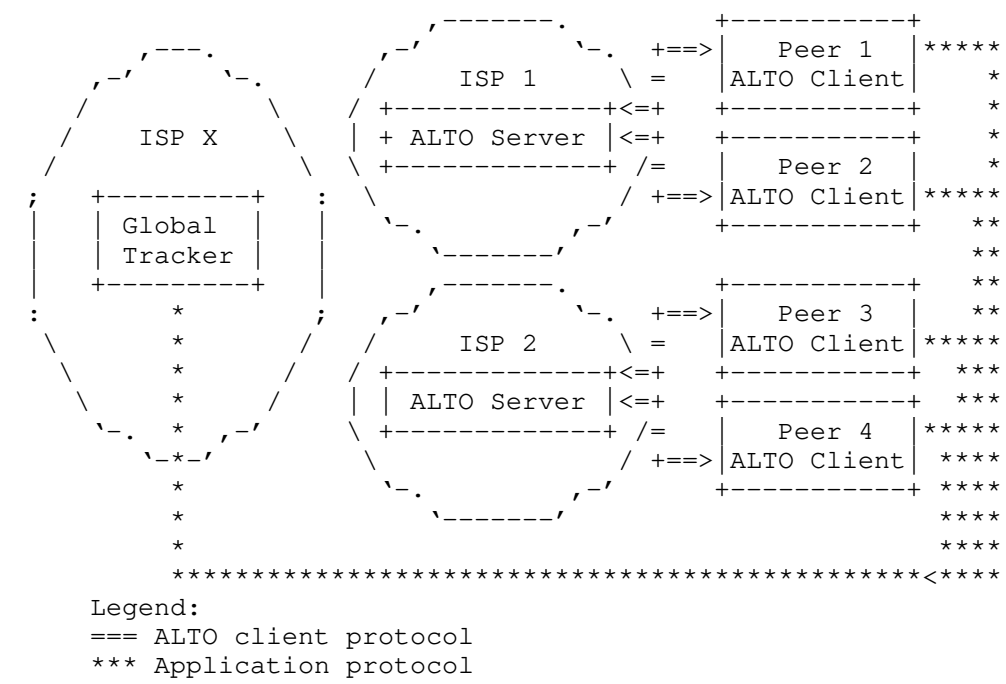


Figure 12: Global Tracker - Local ALTO Servers

The scenario in Figure 12 lets the peers directly communicate with their ISP's ALTO server (i.e., ALTO client embedded in the peers), giving thus the peers the most control on which information they query for, as they can integrate information received from trackers and through direct peer-to-peer knowledge exchange.

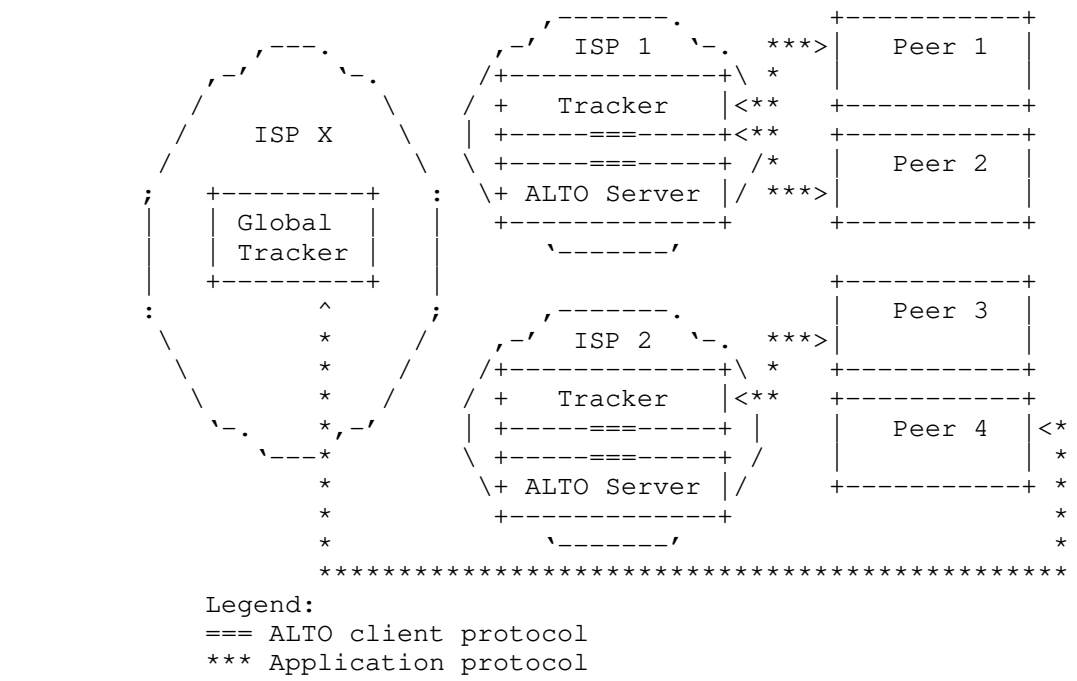


Figure 13: P4P approach with local tracker and local ALTO server

There are some attempts to let ISP's to deploy their own trackers, as shown in Figure 13. In this case, the client has no chance to get guidance from the ALTO server, other than talking to the ISP's tracker. However, the peers would have still chance the contact other trackers, deployed by entities other than the peer's ISP.

Figure 13 and Figure 11 ostensibly take peers the possibility to directly query the ALTO server, if the communication with the ALTO server is not permitted for any reason. However, considering the plethora of different applications of ALTO, e.g., multiple tracker and non-tracker based P2P systems and or applications searching for relays, it seems to be beneficial for all participants to let the peers directly query the ALTO server. The peers are also the single point having all operational knowledge to decide whether to use the ALTO guidance and how to use the ALTO guidance. This is a preference for the scenario depicted in Figure Figure 12.

4.1. Using ALTO for Tracker-based Peer-to-Peer Applications

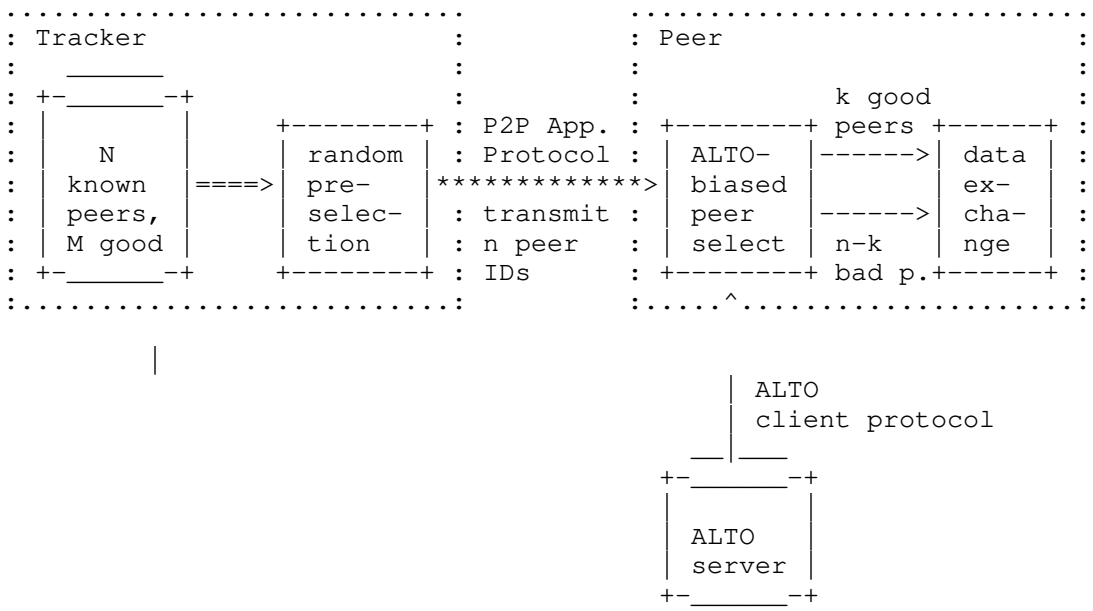


Figure 14: Tracker-based P2P Application with random peer preselection

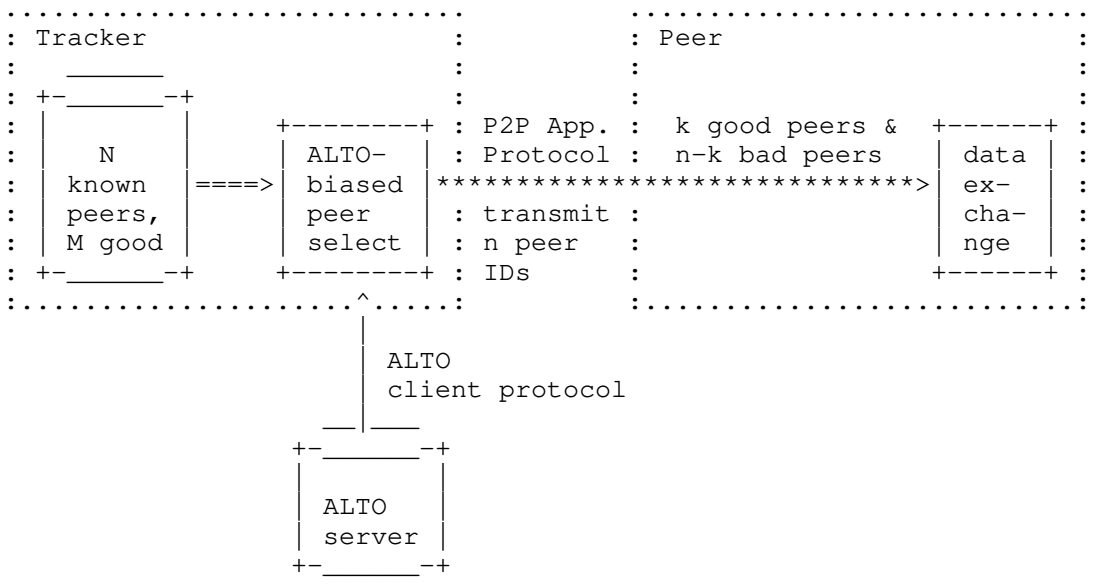


Figure 15: Tracker-based P2P Application with ALTO client in tracker

TBD: explain why Figure 15 usually will yield better results wrt. peer selection than Figure 14.

4.2. Expectations of ALTO

This section hints to some recent experiments conducted with ALTO-like deployments in Internet Service Provider (ISP) network's. NTT performed tests with their HINT server implementation and dummy nodes to gain insight on how an ALTO-like service influence a peer-to-peer systems [I-D.kamei-p2p-experiments-japan]. The results of an early experiment conducted in the Comcast network are documented here[RFC5632]

5. Using ALTO for CDNs

Section 2 discussed the placement and usage of ALTO for P2P systems, but not beyond. This section discuss the usage of ALTO for Content Delivery Networks (CDNs). CDNs are used to bring a service (e.g., a web page, videos, etc) closer to the location of the user - where close refers to shorten the distance between the client and the server in the IP topology. CDNs use several techniques to decide which server is closest to a client requesting a service. One common way to do so, is relying on the DNS system, but there are many other ways, see [RFC3568].

The general issue for CDNs, independent of DNS or HTTP Redirect based approaches (see, for instance, [I-D.penno-alto-cdn]), is that the CDN logic has to match the client's IP address with the closest CDN cache. This matching is not trivial, for instance, in DNS based approaches, where the IP address of the DNS original requester is unknown (see [I-D.vandergaast-edns-client-ip] for a discussion of this and a solution approach).

6. Advanced Features

6.1. Cascading ALTO Servers

The main assumptions of ALTO seems to be each ISP operates its own ALTO server independently, irrespectively of the ISP's situation. This may true for most envisioned deployments of ALTO but there are certain deployments that may have different settings. Figure 16 shows such setting, were for example, a university network is connected to two upstream providers. ISP2 if the national research network and ISP1 is a commercial upstream provider to this university network. The university, as well as ISP1, are operating their own ALTO server. The ALTO clients, located on the peers will contact the ALTO server located at the university.

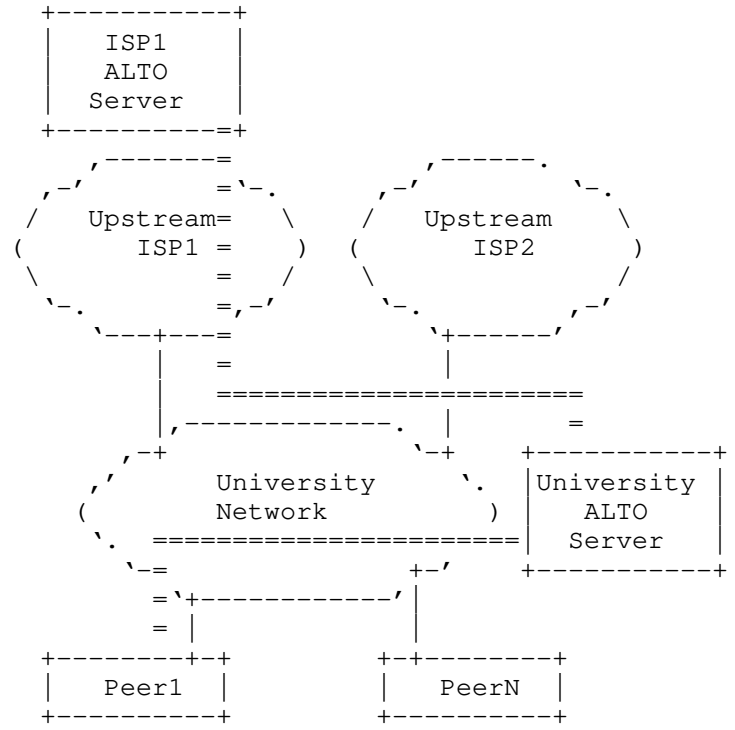


Figure 16: Cascaded ALTO Server

In this setting all "destinations" useful for the peers within ISP2 are free-of-charge for the peers located in the university network (i.e., they are preferred in the rating of the ALTO server). However, all traffic that is not towards ISP2 will be handled by the

ISP1 upstream provider. Therefore, the ALTO server at the university has also to include the guidance given by the ISP1 ALTO server in its replies to the ALTO clients. This can be called cascaded ALTO servers.

6.2. ALTO for IPv4 and IPv6

TBD

6.3. Monitoring ALTO

In addition to providing configuration, an ISP providing ALTO may want to deploy a monitoring infrastructure to assess the benefits of ALTO and adjust its ALTO configuration according to the results of the monitoring.

To construct an effective monitoring infrastructure, the ISP should (1) define the performance metrics to be monitored; (2) and identify and deploy data sources to collect data to compute the performance metrics. We discuss both below.

[Editor's note: Is there a relationship to the IPPM working group at the IETF?]

6.3.1. Monitoring Metrics Definition

- Inter-domain ALTO-Integrated Application Traffic (Network metric): This metric includes total cross domain traffic generated by applications that utilize ALTO guidance. This metric evaluates the impacts of ALTO on the inbound and outbound traffic of a domain.

- Total Inter-domain Traffic (Network metric): This is similar to the preceding but focuses on all of the traffic, ALTO aware or not. One possibility is that some of the reduction of interdomain traffic by ALTO aware applications may (XXX missing words?). This metric is always used with the preceding and the following metrics.

- Intra-domain ALTO-Integrated Application Traffic (Network metric). (XXX description missing)

- Network hop count (Network metric): This metric provides the average number of hops that traffic traverses inside a domain. ALTO may reduce not only traffic volume but also the hops. The metric can also indirectly reflect some application performance (e.g., latency).

- Application download rate (Application metric): This metric measures application performance directly. Download means inbound traffic to one user. Global average means the average value of all users' download rates in one or more domains.

- Application Client type audit(Application metric): this metric gives the audit of client types in ALTO service. The current types include fixed network client and mobile network client.

6.3.2. Monitoring Data Sources

The preceding metrics are derived from data sources. We identify three data sources.

1. Application Log Server: Many application systems deploy Log Servers to collect data.

2. P2P Clients: Some P2P applications may not have Log Servers. When available, P2P client logs can provide data. This is for P2P application

3. OAM: Many ISPs deploy OAM systems to monitor IP layer traffic. An OAM provides traffic monitoring of every network device in its management area. It provides data such as link physical bandwidth and traffic volumes.

6.3.3. Monitoring Structure

As discussed in the preceding section, some data sources are from ISP while some others are from application. When there is a collaboration agreement between the ISP and an application, there can be an integrated monitoring system as shown in the figure below. In particular, an application developer may deploy Monitor Clients to communicate with Monitor Server of the ISP to transmit raw data from the Log Server or P2P clients of the application to the ISP.

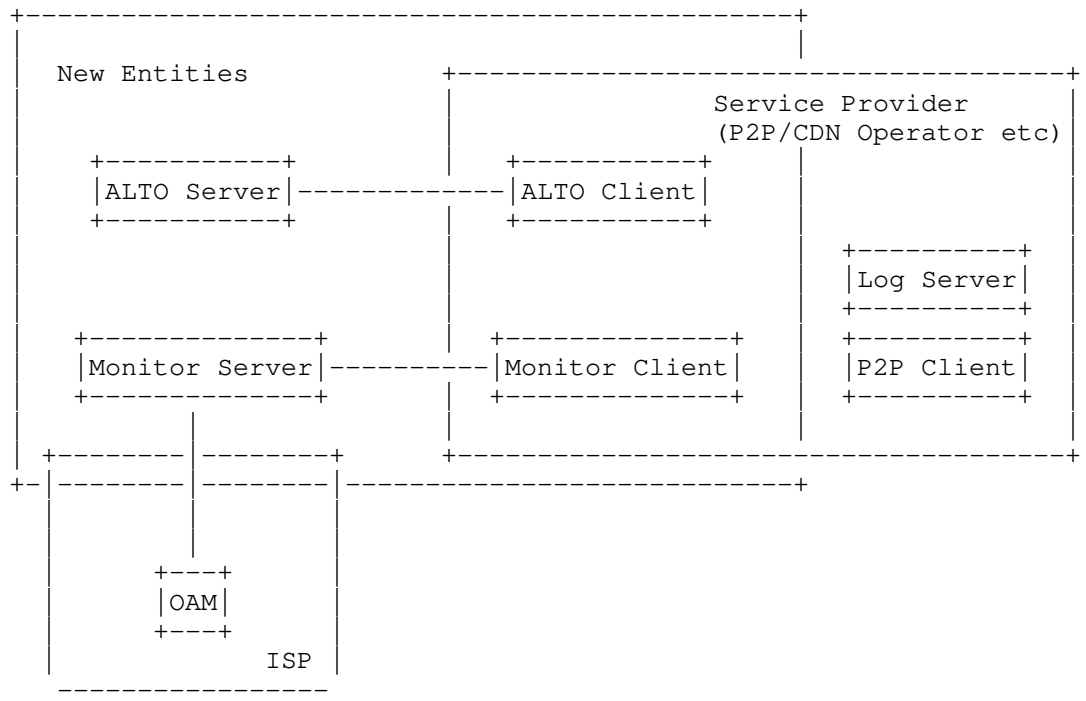


Figure 17: Monitoring Structure

7. Known Limitations of ALTO

This section describes some known limitations of ALTO in general or specific mechanisms in ALTO.

7.1. Limitations of Map-based Approaches

The specification of the ALTO protocol [I-D.ietf-alto-protocol] uses, amongst others mechanism, so-called network maps. The network map approach uses Host Group Descriptors that group one or multiple subnetworks (i.e., IP prefixes) to a single Host Group Descriptor. A set of IP prefixes is called partition and the associated Host Group Descriptor is called partition ID. The "costs" between the various partition IDs is stored in a second map, the cost map. Map-based approaches are chosen as they lower the signaling load on the server, as the maps have only to be retrieved if they are changed.

The main assumption for map-based approaches is that the information provided in these maps is static for a longer period of time, where this period of time refers to days, but not hours or even minutes. This assumption is fine, as long as the network operator does not change any parameter, e.g., routing within the network and to the upstream peers, IP address assignment stays stable (and thus the mapping to the partitions). However, there are several cases where this assumption is not valid, as:

1. ISPs reallocate IPv4 subnets from time to time;

2. ISPs reallocate IPv4 subnets on short notice;

3. IP prefix blocks may be assigned to a single DSLAM which serves a variety of access networks.

For 1): ISPs reallocate IPv4 subnets within their infrastructure from time to time, partly to ensure the efficient usage of IPv4 addresses (a scarce resource), and partly to enable efficient route tables within their network routers. The frequency of these "renumbering events" depend on the growth in number of subscribers and the availability of address space within the ISP. As a result, a subscriber’s household device could retain an IPv4 address for as short as a few minutes, or for months at a time or even longer.

> Some folks have suggested that ISPs providing ALTO services could sub-divide their subscribers’ devices into different IPv4 subnets (or certain IPv4 address ranges) based on the purchased service tier, as well as based on the location in the network topology. The problem is that this sub-allocation of IPv4 subnets tends to decrease the efficiency of IPv4 address allocation. A growing ISP

that needs to maintain high efficiency of IPv4 address utilization may be reluctant to jeopardize their future acquisition of IPv4 address space.

However, this is not an issue for map-based approaches if changes are applied in the order of days.

For 2): ISPs can use techniques, such as ODAP (XXX) that allow the reallocation of IP prefixes on very short notice, i.e., within minutes. An IP prefix that has no IP address assignment to a host anymore can be reallocate to areas where there is currently a high demand for IP addresses.

For 3): In DSL-based access networks, IP prefixes are assigned to DSLAMs which are the first IP-hop in the access-network between the CPE and the Internet. The access-network between CPE and DSLAM (called aggregation network) can have varying characteristics (and thus associated costs), but still using the same IP prefix. For instance one IP addresses IP11 out of a IP prefix IP1 can be assigned to a VDSL (e.g., 2 MBit/s uplink) access-line while the subsequent IP address IP12 is assigned to a slow ADSL line (e.g., 128 kbit/s uplink). These IP addresses are assigned on a first come first served basis, i.e., the a single IP address out of the same IP prefix can change its associated costs quite fast. This may not be an issue with respect to the used upstream provider (thus the cross ISP traffic) but depending on the capacity of the aggregation-network this may raise to an issue.

7.2. Limitiations of Non-Map-based Approaches

The specification of the ALTO protocol [I-D.ietf-alto-protocol] uses, amongst others mechanism, a mechanism called Endpoint Cost Service. ALTO clients can ask guidance for specific IP addresses to the ALTO server. However, asking for IP addresses, asking with long lists of IP addresses, and asking quite frequent may overload the ALTO server. The server has to rank each received IP address which causes load at the server. This may be amplified by the fact that not only a single ALTO client is asking for guidance, but a larger number of them.

Caching of IP addresses at the ALTO client or the usage of the H12 approach [I-D.kiesel-alto-h12] in conjunction with caching may lower the query load on the ALTO server.

7.3. General Challenges

An ALTO server stores information about preferences (e.g., a list of preferred autonomous systems, IP ranges, etc) and ALTO clients can retrieve these preferences. However, there are basically two

different approaches on where the preferences are actually processed:

1. The ALTO server has a list of preferences and clients can retrieve this list via the ALTO protocol. This preference list can be partially updated by the server. The actual processing of the data is done on the client and thus there is no data of the client's operation revealed to the ALTO server .

2. The ALTO server has a list of preferences or preferences calculated during runtime and the ALTO client is sending information of its operation (e.g., a list of IP addresses) to the server. The server is using this operational information to determine its preferences and returns these preferences (e.g., a sorted list of the IP addresses) back to the ALTO client.

Approach 1 (we call it H1) has the advantage (seen from the client) that all operational information stays within the client and is not revealed to the provider of the server. On the other hand, does approach 1 require that the provider of the ALTO server, i.e., the network operator, reveals information about its network structure (e.g., AS numbers, IP ranges, topology information in general) to the ALTO client.

Approach 2 (we call it H2) has the advantage (seen from the operator) that all operational information stays with the ALTO server and is not revealed to the ALTO client. On the other hand, does approach 2 require that the clients send their operational information to the server.

Both approaches have their pros and cons and are extensively discussed on the ALTO mailing list. But there is basically a dilemma: Approach 1 is seen as the only working solution by peer-to-peer software vendors and approach 2 is seen as the only working by the network operators. But neither the software vendors nor the operators seem to willing to change their position. However, there is the need to get both sides on board, to come to a solution.

8. Extensions to the ALTO Protocol

8.1. Host Group Descriptors

Host group descriptors are used in the ALTO client protocol to describe the location of a host in the network topology. The ALTO client protocol specification defines a basic set of host group descriptor types, which have to be supported by all implementations, and an extension procedure for adding new descriptor types . The following list gives an overview on further host group descriptor types that have been proposed in the past, or which are in use by ALTO-related prototype implementations. This list is not intended as normative text. Instead, the only purpose of the following list is to document the descriptor types that have been proposed so far, and to solicit further feedback and discussion:

- Autonomous System (AS) number
- Protocol-specific group identifiers, which expand to a set of IP address ranges (CIDR) and/or AS numbers. In one specific solution proposal, these are called Partition ID (PID).

8.2. Rating Criteria

Rating criteria are used in the ALTO client protocol to express topology- or connectivity-related properties, which are evaluated in order to generate the ALTO guidance. The ALTO client protocol specification defines a basic set of rating criteria, which have to be supported by all implementations, and an extension procedure for adding new criteria . The following list gives an overview on further rating criteria that have been proposed in the past, or which are in use by ALTO-related prototype implementations. This list is not intended as normative text. Instead, the only purpose of the following list is to document the rating criteria that have been proposed so far, and to solicit further feedback and discussion:

8.2.1. Distance-related Rating Criteria

- Relative topological distance: relative means that a larger numerical value means greater distance, but it is up to the ALTO service how to compute the values, and the ALTO client will not be informed about the nature of the information. One way of generating this kind of information MAY be counting AS hops, but when querying this parameter, the ALTO client MUST NOT assume that the numbers actually are AS hops.
- Absolute topological distance, expressed in the number of traversed autonomous systems (AS).

- Absolute topological distance, expressed in the number of router hops (i.e., how much the TTL value of an IP packet will be decreased during transit).

- Absolute physical distance, based on knowledge of the approximate geolocation (continent, country) of an IP address.

8.2.2. Charging-related Rating Criteria

- Traffic volume caps, in case the Internet access of the resource consumer is not charged by "flat rate". For each candidate resource provider, the ALTO service could indicate the amount of data that may be transferred from/to this resource provider until a given point in time, and how much of this amount has already been consumed. Furthermore, it would have to be indicated how excess traffic would be handled (e.g., blocked, throttled, or charged separately at an indicated price). The interaction of several applications running on a host, out of which some use this criterion while others don't, as well as the evaluation of this criterion in resource directories, which issue ALTO queries on behalf of other peers, are for further study.

8.2.3. Performance-related Rating Criteria

The following rating criteria are subject to the remarks below.

- The minimum achievable throughput between the resource consumer and the candidate resource provider, which is considered useful by the application (only in ALTO queries), or

- An arbitrary upper bound for the throughput from/to the candidate resource provider (only in ALTO responses). This may be, but is not necessarily the provisioned access bandwidth of the candidate resource provider.

- The maximum round-trip time (RTT) between resource consumer and the candidate resource provider, which is acceptable for the application for useful communication with the candidate resource provider (only in ALTO queries), or

- An arbitrary lower bound for the RTT between resource consumer and the candidate resource provider (only in ALTO responses). This may be, for example, based on measurements of the propagation delay in a completely unloaded network.

The ALTO client MUST be aware, that with high probability, the actual performance values differ significantly from these upper and lower bounds. In particular, an ALTO client MUST NOT consider the "upper

bound for throughput" parameter as a permission to send data at the indicated rate without using congestion control mechanisms.

The discrepancies are due to various reasons, including, but not limited to the facts that

- the ALTO service is not an admission control system
- the ALTO service may not know the instantaneous congestion status of the network
- the ALTO service may not know all link bandwidths, i.e., where the bottleneck really is, and there may be shared bottlenecks
- the ALTO service may not know whether the candidate peer itself is overloaded
- the ALTO service may not know whether the candidate peer throttles the bandwidth it devotes for the considered application
- the ALTO service may not know whether the candidate peer will throttle the data it sends to us (e.g., because of some fairness algorithm, such as tit-for-tat)

Because of these inaccuracies and the lack of complete, instantaneous state information, which are inherent to the ALTO service, the application must use other mechanisms (such as passive measurements on actual data transmissions) to assess the currently achievable throughput, and it MUST use appropriate congestion control mechanisms in order to avoid a congestion collapse. Nevertheless, these rating criteria may provide a useful shortcut for quickly excluding candidate resource providers from such probing, if it is known in advance that connectivity is in any case worse than what is considered the minimum useful value by the respective application.

8.2.4. Inappropriate Rating Criteria

Rating criteria that SHOULD NOT be defined for and used by the ALTO service include:

- Performance metrics that are closely related to the instantaneous congestion status. The definition of alternate approaches for congestion control is explicitly out of the scope of ALTO. Instead, other appropriate means, such as using TCP based transport, have to be used to avoid congestion.

9. API between ALTO Client and Application

This sections gives some informational guidance on how the interface between the actual application using the ALTO guidance and the ALTO client can look like.

This is still TBD.

10. Security Considerations

The ALTO protocol itself, as well as, the ALTO client and server raise new security issues beyond the one mentioned in [I-D.ietf-alto-protocol] and issues related to message transport over the Internet. For instance, Denial of Service (DoS) is of interest for the ALTO server and also for the ALTO client. A server can get overloaded if too many TCP requests hit the server, or if the query load of the server surpasses the maximum computing capacity. An ALTO client can get overloaded if the responses from the sever are, either intentionally or due to an implementation mistake, too large to be handled by that particular client.

10.1. Information Leakage from the ALTO Server

The ALTO server will be provisioned with information about the owning ISP's network and very likely also with information about neighboring ISPs. This information (e.g., network topology, business relations, etc) is consider to be confidential to the ISP and must not be revealed.

The ALTO server will naturally reveal parts of that information in small doses to peers, as the guidance given will depend on the above mentioned information. This is seen beneficial for both parties, i.e., the ISP's and the peer's. However, there is the chance that one or multiple peers are querying an ALTO server with the goal to gather information about network topology or any other data considered confidential or at least sensitive. It is unclear whether this is a real technical security risk or whether this is more a perceived security risk.

10.2. ALTO Server Access

Depending on the use case of ALTO, several access restrictions to an ALTO server may or may not apply. For an ALTO server that is solely accessible by peers from the ISP network (as shown in Figure 12), for instance, the source IP address can be used to grant only access from that ISP network to the server. This will "limit" the number of peers able to attack the server to the user's of the ISP (however, including botnet computers).

On the other hand, if the ALTO server has to be accessible by parties not located in the ISP's network (see Figure Figure 11), e.g., by a third-party tracker or by a CDN system outside the ISP's network, the access restrictions have to be more loose. In the extreme case, i.e., no access restrictions, each and every host in the Internet can access the ALTO server. This might no the intention of the ISP, as the server is not only subject to more possible attacks, but also on

the load imposed to the server, i.e., possibly more ALTO clients to serve and thus more work load.

10.3. Faking ALTO Guidance

It has not yet been investigated how a faked or wrong ALTO guidance by an ALTO server can impact the operation of the network and also the peers.

Here is a list of examples how the ALTO guidance could be faked and what possible consequences may arise:

Sorting An attacker could change to sorting order of the ALTO guidance (given that the order is of importance, otherwise the ranking mechanism is of interest), i.e., declaring peers located outside the ISP as peers to be preferred. This will not pose a big risk to the network or peers, as it would mimic the "regular" peer operation without traffic localization, apart from the communication/processing overhead for ALTO. However, it could mean that ALTO is reaching the opposite goal of shuffling more data across ISP boundaries, incurring more costs for the ISP.

Preference of a single peer A single IP address (thus a peer) could be marked as to be preferred all over other peers. This peer can be located within the local ISP or also in other parts of the Internet (e.g., a web server). This could lead to the case that quite a number of peers to trying to contact this IP address, possibly causing a Denial of Service (DoS) attack.

This section is solely giving a first shot on security issues related to ALTO deployments.

11. Conclusion

This is the first version of the deployment considerations and for sure the considerations are yet incomplete and imprecise.

12. References

12.1. Normative References

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

[RFC3568] Barbir, A., Cain, B., Nair, R., and O. Spatscheck, "Known Content Network (CN) Request-Routing Mechanisms", RFC 3568, July 2003.

12.2. Informative References

[I-D.ietf-alto-protocol] Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-09 (work in progress), June 2011.

[I-D.ietf-alto-reqs] Kiesel, S., Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", draft-ietf-alto-reqs-10 (work in progress), June 2011.

[I-D.kamei-p2p-experiments-japan] Kamei, S., Momose, T., and T. Inoue, "ALTO-Like Activities and Experiments in P2P Network Experiment Council", draft-kamei-p2p-experiments-japan-05 (work in progress), March 2011.

[I-D.kiesel-alto-3pdisc] Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M., Tomsu, M., and S. Yongchao, "ALTO Server Discovery Protocol", draft-kiesel-alto-3pdisc-05 (work in progress), March 2011.

[I-D.kiesel-alto-h12] Kiesel, S. and M. Stiemerling, "ALTO H12", draft-kiesel-alto-h12-02 (work in progress), March 2010.

[I-D.lee-alto-chinatelecom-trial] Li, K. and G. Jian, "ALTO and DECADE service trial within China Telecom", draft-lee-alto-chinatelecom-trial-02 (work in progress), April 2011.

[I-D.penno-alto-cdn] Penno, R., Medved, J., Alimi, R., Yang, R., and S. Previdi, "ALTO and Content Delivery Networks", draft-penno-alto-cdn-03 (work in progress), March 2011.

[I-D.vandergaast-edns-client-ip]
Contavalli, C., Gaast, W., Leach, S., and D. Rodden, "Client IP information in DNS requests", draft-vandergaast-edns-client-ip-01 (work in progress), May 2010.

[RFC5632] Griffiths, C., Livingood, J., Popkin, L., Woundy, R., and Y. Yang, "Comcast's ISP Experiences in a Proactive Network Provider Participation for P2P (P4P) Technical Trial", RFC 5632, September 2009.

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

Appendix A. Acknowledgments

Xianghui Sun, Lee Kai, and Richard Yang contributed Section 3 and Section 6.3

Martin Stiemerling is partially supported by the COAST project (COntent Aware Searching, retrieval and sTreaming, http://www.coast-fp7.eu), a research project supported by the European Commission under its 7th Framework Program (contract no. 248036). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the COAST project or the European Commission.

Authors' Addresses

Martin Stiemerling
NEC Laboratories Europe
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 4342 113
Fax: +49 6221 4342 155
Email: martin.stiemerling@neclab.eu
URI: http://ietf.stiemerling.org

Sebastian Kiesel
University of Stuttgart, Computing Center
Allmandring 30
Stuttgart 70550
Germany

Email: ietf-alto@skiesel.de

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2011

R. Alimi, Ed.
Google
R. Penno, Ed.
Juniper Networks
Y. Yang, Ed.
Yale University
June 27, 2011

ALTO Protocol
draft-ietf-alto-protocol-09.txt

Abstract

Networking applications today already have access to a great amount of Inter-Provider network topology information. For example, views of the Internet routing table are easily available at looking glass servers and entirely practical to be downloaded by clients. What is missing is knowledge of the underlying network topology from the ISP or Content Provider (henceforth referred as Provider) point of view. In other words, what a Provider prefers in terms of traffic optimization -- 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 [RFC2119].

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 December 29, 2011.

Copyright Notice

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 the 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 this document is to specify a simple and unified protocol that meets the ALTO requirements [I-D.ietf-alto-reqs] while 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

- P4P [I-D.p4p-framework], [P4P-SIGCOMM08], [I-D.wang-alto-p4p-specification];
- ALTO Info-Export [I-D.shalunov-alto-infoexport];
- Query/Response [I-D.saumitra-alto-queryresponse], [I-D.saumitra-alto-multi-ps];
- ATTP [ATTP];
- Proxidor [I-D.akonjang-alto-proxidor].

See Appendix A for a list of people that have contributed significantly to this effort and the projects and proposals listed above.

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 peer selection process in distributed applications in order to increase locality of traffic, improve user-experience, amongst others. It also helps ISPs to efficiently manage traffic that traverses more expensive links such as transit and backup links, thus allowing a better provisioning of the networking infrastructure.

1.3.2. 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.

An example type of application is a Peer-to-Peer overlay where peer selection can be improved by including ALTO information in the selection process.

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. This section first introduces the terminology, and then defines the ALTO architecture and the ALTO Protocol's place in the overall architecture.

2.1. Terminology

We use the following terms defined in [RFC5693]: 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, Autonomous System Number (ASN), and Network Location.

2.1.1. Endpoint Address

An endpoint address represents the communication address of an endpoint. 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.

Each Endpoint Address has an associated Address Type, which indicates both its syntax and semantics.

2.1.2. ASN

An Autonomous System Number.

2.1.3. Network Location

Network Location is a generic term denoting a single endpoint or group of endpoints.

2.1.4. ALTO Information

ALTO Information is a generic term referring to the network information sent by an ALTO Server.

2.1.5. ALTO Information Base

Internal representation of the ALTO Information maintained by the ALTO Server. Note that the structure of this internal representation is not defined by this document.

2.2. ALTO Service and Protocol Scope

An ALTO Server conveys the network information from the perspective of a network region; the ALTO Server presents its "my-Internet View" of the network region. In particular, an ALTO Server defines network Endpoints (and aggregations thereof) and generic costs amongst them, both from the network region's own perspective. A network region in this context can be an Autonomous System, an ISP, or perhaps a smaller region or 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 Server prepares ALTO Information; an ALTO Client uses ALTO Service Discovery to identify an appropriate ALTO Server; 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.

Note that it may also be possible for ALTO Servers to exchange network information with other ALTO Servers (either within the same administrative domain or another administrative domain with the consent of both parties) in order to adjust exported ALTO Information. Such a protocol is also outside the scope of this specification.

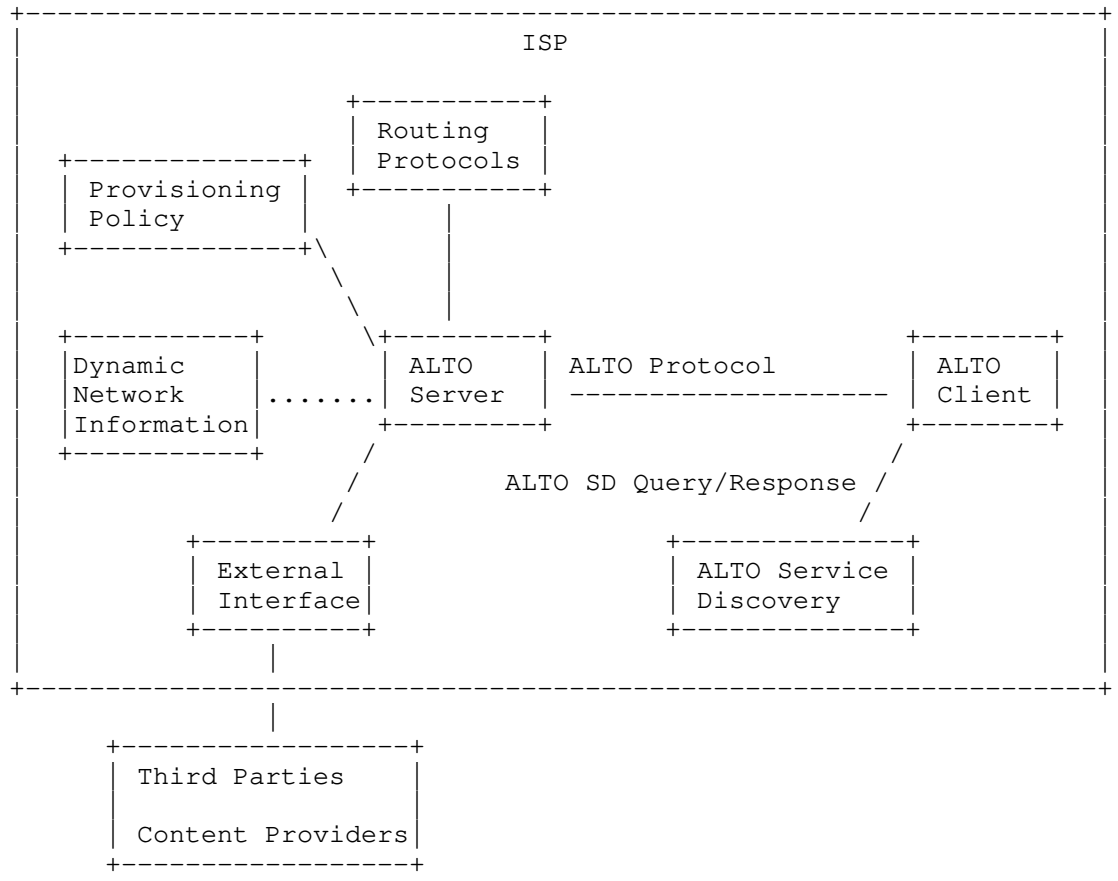


Figure 1: Basic ALTO Architecture.

3. Protocol Structure

The ALTO Protocol uses a simple extensible framework to convey network information. In the general framework, the ALTO protocol will convey properties on both Network Locations and the paths between Network Locations.

In this document, we focus on a particular Endpoint property to denote the location of an endpoint, and provider-defined costs for paths between pairs of Network Locations.

The ALTO Protocol is built on a common transport protocol, messaging structure and encoding, and transaction model. The protocol is subdivided into services of related functionality. ALTO-Core provides the Server Information Service and the Map Service to provide ALTO Information. Other ALTO Information services can provide additional functionality. There are three such services defined in this document: the Map Filtering Service, Endpoint Property Service, and Endpoint Cost Service. Additional services may be defined in companion documents. Note that functionality offered in different services are not totally non-overlapping (e.g., the Map Service and Map Filtering Service).

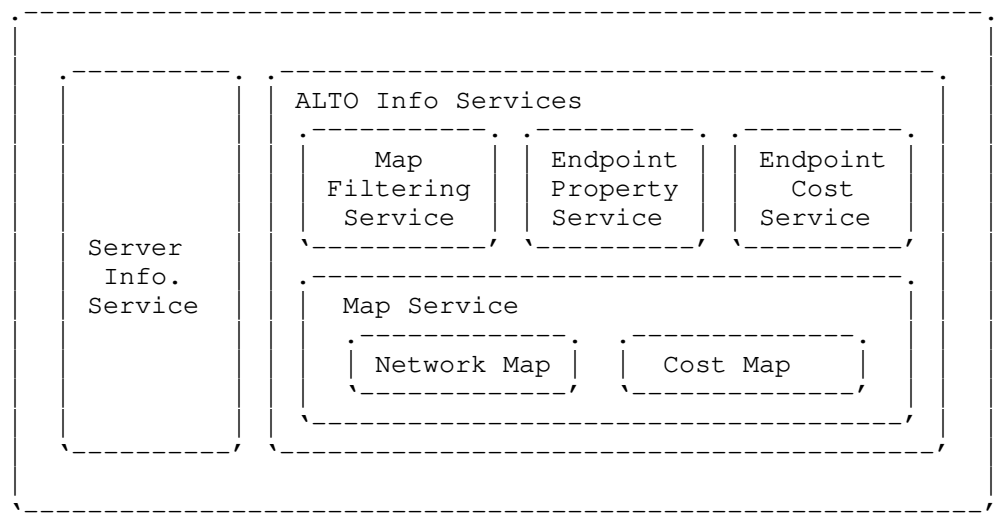


Figure 2: ALTO Protocol Structure

3.1. Server Information Service

The Server Information Service lists the details on the information that can be provided by an ALTO Server and perhaps other ALTO Servers maintained by the network provider. The configuration includes, for example, details about the operations and cost metrics supported by the ALTO Server and other related ALTO Servers that may be usable by an ALTO Client.

3.2. ALTO Information Services

Multiple, distinct services are defined to allow ALTO Clients to query ALTO Information from an ALTO Server. The ALTO Server internally maintains an ALTO Information Base that encodes the network provider's preferences. The ALTO Information Base encodes the Network Locations defined by the ALTO Server (and their corresponding properties), as well as the provider-defined costs between pairs of Network Locations.

3.2.1. Map Service

The Map Service provides batch information to ALTO Clients in the form of Network Map and Cost Map. The Network Map (See Section 4) provides the full set of Network Location groupings defined by the ALTO Server and the endpoints contained with each grouping. The Cost Map (see Section 5) provides costs between the defined groupings.

These two maps can be thought of (and implemented as) as simple files with appropriate encoding provided by the ALTO Server.

3.2.2. Map Filtering Service

Resource constrained ALTO Clients may benefit from query results being filtered at the ALTO Server. This avoids an ALTO Client spending network bandwidth or CPU collecting results and performing client-side filtering. The Map Filtering Service allows ALTO Clients to query for the ALTO Server Network Map and Cost Map based on additional parameters.

3.2.3. Endpoint Property Service

This service allows ALTO Clients to look up properties for individual endpoints. An example endpoint property is its Network Location (its grouping defined by the ALTO Server) or connectivity type (e.g., ADSL, Cable, or FTTH).

3.2.4. Endpoint Cost Service

Some ALTO Clients may also benefit from querying for costs and rankings based on endpoints. The Endpoint Cost Service allows an ALTO Server to return either numerical costs or ordinal costs (rankings) directly amongst Endpoints.

4. Network Map

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 losing 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 definition of proximity varies depending on the granularity of the ALTO information configured by the provider. 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.

As used in this document, the Network Map refers to the syntax and semantics of the information distributed by the ALTO Server. This document does not discuss the internal representation of this data structure within the ALTO Server.

4.1. PID

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

A PID is an identifier that provides an indirect and network-agnostic way to specify an aggregation of network endpoints that may be treated similarly, based on network topology, type, or other properties. 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 and can improve scalability. In particular, network preferences (costs) may be specified between PIDs, allowing cost information to be more compactly represented and updated at a faster time scale than the network aggregations themselves.

Using PIDs, the Network Map may also be used to communicate simple preferences with only minimal information from the Cost Map. 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. The ISP may aggregate endhosts within a PoP into a single PID in the Network Map. The Cost Map may be encoded to indicate that peering within the same PID is preferred; for example, cost(PID_i, PID_i) == c* and cost(PID_i, PID_j) > c* for i != j. Section 5 provides further details about Cost Map structure.

4.2. Endpoint Addresses

Communicating endpoints may have many types of addresses, such as IP addresses, MAC addresses, or overlay IDs. The current specification only considers IP addresses.

4.2.1. IP Addresses

The endpoints aggregated into a PID are denoted by a list of IP prefixes. When either an ALTO Client or ALTO Server needs to determine which PID in a Network Map contains a particular IP address, longest-prefix matching MUST be used.

A Network Map MUST define a PID for each possible address in the IP address space for all of the address types contained in the map. A RECOMMENDED way to satisfy this property is to define a PID that contains the 0.0.0.0/0 prefix for IPv4 or ::/0 (for IPv6).

Each endpoint MUST map into exactly one PID. Since longest-prefix matching is used to map an endpoint to a PID, this can be accomplished by ensuring that no two PIDs contain an identical IP prefix.

4.3. Example Network Map

Figure 3 illustrates an example Network Map. PIDs are used to identify network-agnostic aggregations.

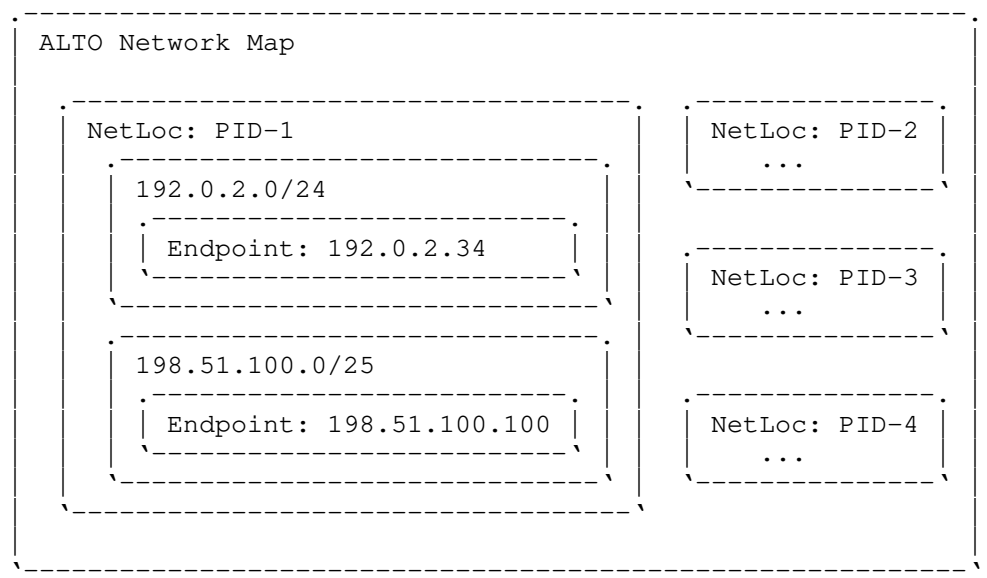


Figure 3: Example Network Map

5. Cost Map

An ALTO Server indicates preferences amongst network locations in the form of Path Costs. Path Costs are generic costs and can be internally computed by a network provider according to its own needs.

An ALTO Cost Map defines Path Costs pairwise amongst sets of source and destination Network Locations.

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.

As used in this document, the Cost Map refers to the syntax and semantics of the information distributed by the ALTO Server. This document does not discuss the internal representation of this data structure within the ALTO Server.

5.1. Cost Attributes

Path Costs have attributes:

- Type: identifies what the costs represent;
- Mode: identifies how the costs should be interpreted.

Certain queries for Cost Maps allow the ALTO Client to indicate the desired Type and Mode.

5.1.1. 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 strings.

5.1.1.1. Cost Type: routingcost

An ALTO Server MUST define the 'routingcost' Cost Type.

This Cost Type conveys a generic measure for the cost of routing traffic from a source to a destination. Lower values indicate a higher preference for traffic to be sent from a source to a destination.

Note that an ISP may internally compute routing cost using any method it chooses (e.g., air-miles or hop-count) as long as it conforms to these semantics.

5.1.2. Cost Mode

The Mode attribute indicates how costs should be interpreted. Specifically, the Mode attribute indicates whether returned costs 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 handle such costs differently.

Cost Modes are indicated in protocol messages as strings.

An ALTO Server MUST support at least one of 'numerical' and 'ordinal' costs. ALTO Clients SHOULD be cognizant of operations when a desired cost mode is not supported. For example, an ALTO Client desiring numerical costs may adjust behavior if only the ordinal Cost Mode is

available. Alternatively, an ALTO Client desiring ordinal costs may construct ordinal costs given numerical values if only the numerical Cost Mode is available.

5.1.2.1. Cost Mode: numerical

This Cost Mode is indicated by the string 'numerical'. This mode indicates that it is safe to perform numerical operations (e.g. summation) on the returned costs.

5.1.2.2. Cost Mode: ordinal

This Cost Mode is indicated by the string 'ordinal'. This mode indicates that the costs values to a set of Destination Network Locations from a particular Source Network Location are a ranking, with lower values indicating a higher preference. The values are non-negative integers. Ordinal cost values from a particular Source Network Location to a set of Destination Network Locations need not be unique nor contiguous. In particular, from the perspective of a particular Source Network Location, two Destination Network Locations may have an identical rank (ordinal cost value). This document does not specify any behavior by an ALTO Client in this case; an ALTO Client may decide to break ties by random selection, other application knowledge, or some other means.

It is important to note that the values in the Cost Map provided with the ordinal Cost Mode are not necessarily the actual cost known to the ALTO Server.

5.2. Cost Map Structure

A query for a Cost Map either explicitly or implicitly includes a list of Source Network Locations and a list of Destination Network Locations. (Recall that a Network Location can be an endpoint address or a PID.)

Specifically, assume that a 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].

The ALTO Server will return 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). We refer to this structure as a Cost Map.

If the Cost Mode is 'ordinal', the Path Cost of each communicating pair is relative to the m*n entries.

5.3. Network Map and Cost Map Dependency

If a Cost Map contains PIDs in the list of Source Network Locations or the list of Destination Network Locations, the Path Costs are generated based on a particular Network Map (which defines the PIDs). Version Tags are introduced to ensure that ALTO Clients are able to use consistent information even though the information is provided in two maps.

A Version Tag is an opaque string associated with a Network Map maintained by the ALTO Server. When the Network Map changes, the Version Tag MUST also be changed. (Thus, the Version Tag is defined similarly to HTTP's Entity Tags; see Section 3.11 of [RFC2616].) Possibilities for generating a Version Tag include the last-modified timestamp for the Network Map, or a hash of its contents.

A Network Map distributed by the ALTO Server includes its Version Tag. A Cost Map referring to PIDs also includes the Version Tag of the Network Map on which it is based.

6. Protocol Design Overview

The ALTO Protocol design uses a REST-ful design with the goal of leveraging current HTTP [RFC2616] implementations and infrastructure. The REST-ful design supports flexible deployment strategies and provides extensibility. ALTO requests and responses are encoded with JSON [RFC4627].

6.1. Benefits

Benefits enabled by these design choices include easier understanding and debugging, mature libraries, tools, infrastructure, and caching and redistribution of ALTO information for increased scalability.

6.1.1. Existing Infrastructure

HTTP is a natural choice for integration with existing applications and infrastructure. In particular, the ALTO Protocol design leverages:

- the huge installed base of infrastructure, including HTTP caches,
- mature software implementations,
- the fact that many P2P clients already have an embedded HTTP client, and

- authentication and encryption mechanisms in HTTP and SSL/TLS.

6.1.2. ALTO Information Reuse and Redistribution

ALTO information may be useful to a large number of applications and users. For example, an identical Network Map may be used by all ALTO Clients querying a particular ALTO Server. At the same time, distributing ALTO information must be efficient and not become a bottleneck.

Beyond integration with existing HTTP caching infrastructure, ALTO information may also be cached or redistributed using application-dependent mechanisms, such as P2P DHTs or P2P file-sharing. This document does not define particular mechanisms for such redistribution, but it does define the primitives (e.g., digital signatures) needed to support such a mechanism. See [I-D.gu-alto-redistribution] for further discussion.

Note that if caching or redistribution is used, the response message may be returned from another (possibly third-party) entity. Reuse and Redistribution is further discussed in Section 12.4. Protocol support for redistribution is specified in Section 8.

6.2. Protocol Design

The ALTO Protocol uses a REST-ful design. There are two primary components to this design:

- Information Resources: Each service provides network information as a set of resources, which are distinguished by their media types [RFC2046]. An ALTO Client may construct an HTTP request for a particular resource (including any parameters, if necessary), and an ALTO Server returns the requested resource in an HTTP response.

- Information Resource Directory: An ALTO Server provides to ALTO Clients a list of available resources and the URI at which each is provided. This document refers to this list as the Information Resource Directory. This directory is the single entry point to an ALTO Service. ALTO Clients consult the directory to determine the services provided by an ALTO Server.

7. Protocol Specification

This section first specifies general client and server processing, followed by a detailed specification for each ALTO Information Resource.

7.1. Notation

This document uses an adaptation of the C-style struct notation to define the required and optional members of JSON objects. Unless explicitly noted, each member of a struct is REQUIRED.

The types 'JSONString', 'JSONNumber', 'JSONBool' indicate the JSON string, number, and boolean types, respectively.

Note that no standard, machine-readable interface definition or schema is provided. Extension documents may document these as necessary.

7.2. Basic Operation

The ALTO Protocol employs standard HTTP [RFC2616]. It is used for discovering available Information Resources at an ALTO Server and retrieving Information Resources. ALTO Clients and ALTO Servers use HTTP requests and responses carrying ALTO-specific content with encoding as specified in this document, and MUST be compliant with [RFC2616].

7.2.1. Discovering Information Resources

To discover available resources, an ALTO Client may request the Information Resource Directory, which an ALTO Server provides at the URI found by the ALTO Discovery protocol.

Informally, an Information Resource Directory enumerates URIs at which an ALTO Server offers Information Resources. Each entry in the directory indicates a URI at which an ALTO Server accepts requests, and returns either the requested Information Resource or an Information Resource Directory that references additional Information Resources. See Section 7.6 for a detailed specification.

7.2.2. Requesting Information Resources

Through the retrieved Information Resource Directories, an ALTO Client can determine whether an ALTO Server supports the desired Information Resource, and if it is supported, the URI at which it is available.

Where possible, the ALTO Protocol uses the HTTP GET method to request resources. However, some ALTO services provide Information Resources that are the function of one or more input parameters. Input parameters are encoded in the HTTP request's entity body, and the request uses the HTTP POST method.

Note that it is possible for an ALTO Server to employ caching for the response to a POST request. This can be accomplished by returning an HTTP 303 status code ("See Other") indicating to the ALTO Client that the resulting Cost Map is available via a GET request to an alternate URL (which may be cached).

When requesting an ALTO Information Resource that requires input parameters specified in a HTTP POST request, an ALTO Client MUST set the Content-Type HTTP header to the media type corresponding to the format of the supplied input parameters.

7.2.3. Response

Upon receiving a request, an ALTO server either returns the requested resource, provides the ALTO Client an Information Resource Directory indicating how to reach the desired resource, or returns an error.

The type of response MUST be indicated by the media type attached to the response (the Content-Type HTTP header). If an ALTO Client receives an Information Resource Directory, it can consult the received directory to determine if any of the offered URIs contain the desired Information Resource.

The generic encoding for an Information Resource is specified in Section 7.3.

Errors are indicated via either ALTO-level error codes, or via HTTP status codes; see Section 7.4.

7.2.4. Client Behavior

7.2.4.1. Using Information Resources

This specification does not indicate any required actions taken by ALTO Clients upon successfully receiving an Information Resource from an ALTO Server. Although ALTO Clients are suggested to interpret the received ALTO Information and adapt application behavior, ALTO Clients are not required to do so.

7.2.4.2. Error Conditions

If an ALTO Client does not successfully receive a desired Information Resource from a particular ALTO Server, it can either choose another server (if one is available) or fall back to a default behavior (e.g., perform peer selection without the use of ALTO information). An ALTO Client may also retry the request at a later time.

7.2.5. Authentication and Encryption

An ALTO Server MAY support SSL/TLS to implement server and/or client authentication, as well as encryption. See [RFC6125] for considerations regarding verification of server identity.

7.2.6. HTTP Cookies

If cookies are included in an HTTP request received by an ALTO Server, they MUST be ignored.

7.2.7. Parsing

This document only details object members used by this specification. Extensions may include additional members within JSON objects defined in this document. ALTO implementations MUST ignore such unknown fields when processing ALTO messages.

7.3. Information Resource

An Information Resource is an HTTP entity body received by an ALTO Server that encodes the ALTO Information desired by an ALTO Client.

This document specifies multiple Information Resources that can be provided by an ALTO Server. Each Information Resource has certain attributes associated with it, indicating its data format, the input parameters it supports, and format of the input parameters.

7.3.1. Capabilities

An ALTO Server may advertise to an ALTO Client that it supports certain capabilities in requests for an Information Resource. For example, if an ALTO Server allows requests for a Cost Map to include constraints, it may advertise that it supports this capability.

7.3.2. Input Parameters Media Type

An ALTO Server may allow an ALTO Client to supply input parameters when requesting certain Information Resources. The format of the input parameters (i.e., as contained in the entity body of the HTTP POST request) is indicated by the media type [RFC2046].

7.3.3. Media Type

The media type [RFC2046] uniquely indicates the data format of the Information Resource as returned by an ALTO Server in the HTTP entity body.

7.3.4. Encoding

Though each Information Resource may have a distinct syntax, they are designed to have a common structure containing generic ALTO-layer metadata about the resource, as well as data itself.

An Information Resource has a single top-level JSON object of type InfoResourceEntity:

```
object {
  InfoResourceMetaData   meta;
  [InfoResourceDataType] data;
} InfoResourceEntity;
```

with members:

meta meta-information pertaining to the Information Resource

data the data contained in the Information Resource

7.3.4.1. Meta Information

Meta information is encoded as a JSON object with type InfoResourceMetaData:

```
object {
  InfoResourceRedistDesc redistribution;    [OPTIONAL]
} InfoResourceMetaData;
```

with members:

redistribution Additional data for use in Information Resources that may be redistributed amongst ALTO Clients. See Section 8.

7.3.4.2. ALTO Information

The "data" member of the InfoResourceEntity encodes the resource-specific data; the structure of this member is detailed later in this section for each particular Information Resource.

7.3.4.3. Signature

An ALTO Server MAY additionally supply a signature asserting that it generated a particular response. See Section 8.2.2.

7.3.4.4. Example

The following is an example of the encoding for an Information Resource:

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {
    "redistribution" : { ... }
  },
  "data" : {
    ...
  }
}
```

7.4. ALTO Errors

If there is an error processing a request, an ALTO Server SHOULD return additional ALTO-layer information, if it is available, in the form of an ALTO Error Resource encoded in the HTTP response's entity body.

If no ALTO-layer information is available, an ALTO Server may omit an ALTO Error resource from the response. An appropriate HTTP status code MUST be set.

It is important to note that the HTTP Status Code and ALTO Error Code have distinct roles. An ALTO Error Code provides detailed information about the why a particular request for an ALTO Resource was not successful. The HTTP status code indicates to HTTP processing elements (e.g., intermediaries and clients) how the response should be treated.

7.4.1. Media Type

The media type for an Error Resource is "application/alto-error+json".

7.4.2. Resource Format

An Error Resource has the format:

```
object {
  JSONString code;
  JSONString reason;    [OPTIONAL]
} ErrorResourceEntity;
```

where:

code An ALTO Error Code defined in Table 1

reason A (free-form) human-readable explanation of the particular error

7.4.3. Error Codes

This document defines ALTO Error Codes to support the error conditions needed for purposes of this document. Additional status codes may be defined in companion or extension documents.

The HTTP status codes corresponding to each ALTO Error Code are defined to provide correct behavior with HTTP intermediaries and clients. When an ALTO Server returns a particular ALTO Error Code, it MUST indicate one of the corresponding HTTP status codes in Table 1in the HTTP response.

If multiple errors are present in a single request (e.g., a request uses a JSONString when a JSONInteger is expected and a required field is missing), then the ALTO Server MUST return exactly one of the detected errors. However, the reported error is implementation defined, since specifying a particular order for message processing encroaches needlessly on implementation technique.

ALTO Error Code	HTTP Status Code(s)	Description
E_SYNTAX	400	Parsing error in request (including identifiers)
E_JSON_FIELD_MISSING	400	Required field missing
E_JSON_VALUE_TYPE	400	JSON Value of unexpected type
E_INVALID_COST_MODE	400	Invalid cost mode
E_INVALID_COST_TYPE	400	Invalid cost type

E_INVALID_PROPERTY_TYPE	400	Invalid property type

Table 1: Defined ALTO Error Codes

7.5. ALTO Types

This section details the format for particular data values used in the ALTO Protocol.

7.5.1. PID Name

A PID Name is encoded as a US-ASCII string. The string MUST be no more than 64 characters, and MUST NOT contain characters other than alphanumeric characters or the '.' separator. The '.' separator is reserved for future use and MUST NOT be used unless specifically indicated by a companion or extension document.

The type 'PIDName' is used in this document to indicate a string of this format.

7.5.2. Endpoints

This section defines formats used to encode addresses for Endpoints. In a case that multiple textual representations encode the same Endpoint address or prefix (within the guidelines outlined in this document), the ALTO Protocol does not require ALTO Clients or ALTO Servers to use a particular textual representation, nor does it require that ALTO Servers reply to requests using the same textual representation used by requesting ALTO Clients. ALTO Clients must be cognizant of this.

7.5.2.1. Address Type

Address Types are encoded as US-ASCII strings consisting of only alphanumeric characters. This document defines the address type "ipv4" to refer to IPv4 addresses, and "ipv6" to refer to IPv6 addresses. Extension documents may define additional Address Types.

The type 'AddressType' is used in this document to indicate a string of this format.

7.5.2.2. Endpoint Address

Endpoint Addresses are encoded as US-ASCII strings. The exact characters and format depend on the type of endpoint address.

The type 'EndpointAddr' is used in this document to indicate a string

of this format.

7.5.2.2.1. IPv4

IPv4 Endpoint Addresses are encoded as specified by the 'IPv4address' rule in Section 3.2.2 of [RFC3986].

7.5.2.2.2. IPv6

IPv6 Endpoint Addresses are encoded as specified in Section 4 of [RFC5952].

7.5.2.2.3. Typed Endpoint Addresses

When an Endpoint Address is used, an ALTO implementation must be able to determine its type. For this purpose, the ALTO Protocol allows endpoint addresses to also explicitly indicate their type.

Typed Endpoint Addresses are encoded as US-ASCII strings of the format 'AddressType:EndpointAddr' (with the ':' character as a separator). The type 'TypedEndpointAddr' is used to indicate a string of this format.

7.5.2.3. Endpoint Prefixes

For efficiency, it is useful to denote a set of Endpoint Addresses using a special notation (if one exists). This specification makes use of the prefix notations for both IPv4 and IPv6 for this purpose.

Endpoint Prefixes are encoded as US-ASCII strings. The exact characters and format depend on the type of endpoint address.

The type 'EndpointPrefix' is used in this document to indicate a string of this format.

7.5.2.3.1. IPv4

IPv4 Endpoint Prefixes are encoded as specified in Section 3.1 of [RFC4632].

7.5.2.3.2. IPv6

IPv6 Endpoint Prefixes are encoded as specified in Section 7 of [RFC5952].

7.5.2.4. Endpoint Address Group

The ALTO Protocol includes messages that specify potentially large sets of endpoint addresses. Endpoint Address Groups provide a more efficient way to encode such sets, even when the set contains endpoint addresses of different types.

An Endpoint Address Group is defined as:

```
object {
  EndpointPrefix [AddressType]<0..*>;
  ...
} EndpointAddrGroup;
```

In particular, an Endpoint Address Group is a JSON object with the name of each member being the string corresponding to the address type, and the member's corresponding value being a list of prefixes of addresses of that type.

The following is an example with both IPv4 and IPv6 endpoint addresses:

```
{
  "ipv4": [
    "192.0.2.0/24",
    "198.51.100.0/25"
  ],
  "ipv6": [
    "2001:db8:0:1::/64",
    "2001:db8:0:2::/64"
  ]
}
```

7.5.3. Cost Mode

A Cost Mode is encoded as a US-ASCII string. The string MUST either have the value 'numerical' or 'ordinal'.

The type 'CostMode' is used in this document to indicate a string of this format.

7.5.4. Cost Type

A Cost Type is encoded as a US-ASCII string. The string MUST be no more than 32 characters, and MUST NOT contain characters other than alphanumeric characters, the hyphen ('-'), or the ':' separator.

Identifiers prefixed with 'priv:' are reserved for Private Use [RFC5226]. Identifiers prefixed with 'exp:' are reserved for Experimental use. All other identifiers appearing in an HTTP request or response with an 'application/alto-*' media type MUST be registered in the ALTO Cost Types registry Section 11.2.

The type 'CostType' is used in this document to indicate a string of this format.

7.5.5. Endpoint Property

An Endpoint Property is encoded as a US-ASCII string. The string MUST be no more than 32 characters, and MUST NOT contain characters other than alphanumeric characters, the hyphen ('-'), or the ':' separator.

Identifiers prefixed with 'priv:' are reserved for Private Use [RFC5226]. Identifiers prefixed with 'exp:' are reserved for Experimental use. All other identifiers appearing in an HTTP request or response with an 'application/alto-*' media type MUST be registered in the ALTO Endpoint Property registry Section 11.3.

The type 'EndpointProperty' is used in this document to indicate a string of this format.

7.6. Information Resource Directory

An Information Resource Directory indicates to ALTO Clients which Information Resources are made available by an ALTO Server.

Since resource selection happens after consumption of the Information Resource Directory, the format of the Information Resource Directory is designed to be simple with the intention of future ALTO Protocol versions maintaining backwards compatibility. Future extensions or versions of the ALTO Protocol SHOULD be accomplished by extending existing media types or adding new media types, but retaining the same format for the Information Resource Directory.

An ALTO Server MUST make an Information Resource Directory available via the HTTP GET method to a URI discoverable by an ALTO Client. Discovery of this URI is out of scope of this document, but could be accomplished by manual configuration or by returning the URI of an

Information Resource Directory from the ALTO Discovery Protocol [I-D.ietf-alto-server-discovery].

7.6.1. Media Type

The media type is "application/alto-directory+json".

7.6.2. Encoding

An Information Resource Directory is a JSON object of type InfoResourceDirectory:

```
object {
  ...
} Capabilities;

object {
  JSONString   uri;
  JSONString   media-types<1..*>;
  JSONString   accepts<0..*>;        [OPTIONAL]
  Capabilities capabilities;         [OPTIONAL]
} ResourceEntry;

object {
  ResourceEntry resources<0..*>;
} InfoResourceDirectory;
```

where the "resources" array indicates a list of Information Resources provided by an ALTO Server. Note that the list of available resources is enclosed in a JSON object for extensibility; future protocol versions may specify additional members in the InfoResourceDirectory object.

Each entry MUST indicate a URI that either directly provides the indicated Information Resource, or responds to a HTTP OPTIONS request which provides an Information Resource Directory with entries of additional Information Resources.

If an ALTO Client makes a GET or POST request to a URI that does not directly provide an indicated Information Resource, the ALTO Server MUST either reply with an HTTP 300 status code ("Multiple Choices") and an Information Resource Directory in the HTTP response's entity body, or indicate an appropriate HTTP status code. Note that in general, it is preferred that ALTO Clients use HTTP OPTIONS requests to discover additional Information Resources.

A URI that directly provides an Information Resource MAY also respond to HTTP OPTIONS requests, but it is not required to do so (in which case, it MUST respond with HTTP 405 status code ("Method Not Allowed"). This allows certain Information Resources to be configured as static files with minimal configuration on some HTTP servers.

Each entry in the directory specifies:

uri A URI at which the ALTO Server provides one or more Information Resources, or an Information Resource Directory indicating additional Information Resources.

media-types The list of all media types of Information Resources (see Section 7.3.3) available via GET or POST requests to the corresponding URI or URIs discoverable via the URI.

accepts The list of all media types of input parameters (see Section 7.3.2) accepted by POST requests to the corresponding URI or URIs discoverable via the URI. If this member is not present, it MUST be assumed to be an empty array.

capabilities A JSON Object enumerating capabilities of an ALTO Server in providing the Information Resource at the corresponding URI and Information Resources discoverable via the URI. If this member is not present, it MUST be assumed to be an empty array. If a capability for one of the offered Information Resources is not explicitly listed here, an ALTO Client may either issue an OPTIONS HTTP request to the corresponding URI to determine if the capability is supported, or assume its default value.

If an entry has an empty list for "accepts", then the corresponding URI MUST support GET requests. If an entry has a non-empty list for "accepts", then the corresponding URI MUST support POST requests. If an ALTO Server wishes to support both GET and POST on a single URI, it MUST specify two entries in the Information Resource Directory.

7.6.3. Example

The following is an example Information Resource Directory returned by an ALTO Server. In this example, the ALTO Server provides additional Network and Cost Maps via a separate subdomain, "custom.alto.example.com". The maps available via this subdomain are Filtered Network and Cost Maps as well as pre-generated maps for the "hopcount" and "routingcost" Cost Types in the "ordinal" Cost Mode.

An ALTO Client can discover the maps available by "custom.alto.example.com" by successfully performing an OPTIONS

request to "http://custom.alto.example.com/maps".

```
GET /directory HTTP/1.1
Host: alto.example.com
Accept: application/alto-directory+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-directory+json

{
  "resources" : [
    {
      "uri" : "http://alto.example.com/serverinfo",
      "media-types" : [ "application/alto-serverinfo+json" ]
    }, {
      "uri" : "http://alto.example.com/networkmap",
      "media-types" : [ "application/alto-networkmap+json" ]
    }, {
      "uri" : "http://alto.example.com/costmap/num/routingcost",
      "media-types" : [ "application/alto-costmap+json" ],
      "capabilities" : {
        "cost-modes" : [ "numerical" ],
        "cost-types" : [ "routingcost" ]
      }
    }, {
      "uri" : "http://alto.example.com/costmap/num/hopcount",
      "media-types" : [ "application/alto-costmap+json" ],
      "capabilities" : {
        "cost-modes" : [ "numerical" ],
        "cost-types" : [ "hopcount" ]
      }
    }, {
      "uri" : "http://custom.alto.example.com/maps",
      "media-types" : [
        "application/alto-networkmap+json",
        "application/alto-costmap+json"
      ],
      "accepts" : [
        "application/alto-networkmapfilter+json",
        "application/alto-costmapfilter+json"
      ]
    }, {
      "uri" : "http://alto.example.com/endpointprop/lookup",
```

```
        "media-types" : [ "application/alto-endpointprop+json" ],
        "accepts" : [ "application/alto-endpointpropparams+json" ],
        "capabilities" : {
          "prop-types" : [ "pid" ]
        }
      }, {
        "uri" : "http://alto.example.com/endpointcost/lookup",
        "media-types" : [ "application/alto-endpointcost+json" ],
        "accepts" : [ "application/alto-endpointcostparams+json" ],
        "capabilities" : {
          "cost-constraints" : true,
          "cost-modes" : [ "ordinal", "numerical" ],
          "cost-types" : [ "routingcost", "hopcount" ]
        }
      }
    ]
  }
```

```
  OPTIONS /maps HTTP/1.1
  Host: custom.alto.example.com
  Accept: application/alto-directory+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-directory+json

{
  "resources" : [
    {
      "uri" : "http://custom.alto.example.com/networkmap/filtered",
      "media-types" : [ "application/alto-networkmap+json" ],
      "accepts" : [ "application/alto-networkmapfilter+json" ]
    }, {
      "uri" : "http://custom.alto.example.com/costmap/filtered",
      "media-types" : [ "application/alto-costmap+json" ],
      "accepts" : [ "application/alto-costmapfilter+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "cost-modes" : [ "ordinal", "numerical" ],
        "cost-types" : [ "routingcost", "hopcount" ]
      }
    }, {
      "uri" : "http://custom.alto.example.com/ord/routingcost",
      "media-types" : [ "application/alto-costmap+json" ],
      "capabilities" : {
        "cost-modes" : [ "ordinal" ],
        "cost-types" : [ "routingcost" ]
      }
    }, {
      "uri" : "http://custom.alto.example.com/ord/hopcount",
      "media-types" : [ "application/alto-costmap+json" ],
      "capabilities" : {
        "cost-modes" : [ "ordinal" ],
        "cost-types" : [ "hopcount" ]
      }
    }
  ]
}
```

7.6.4. Usage Considerations

7.6.4.1. ALTO Client

This document specifies no requirements or constraints on ALTO Clients with regards to how they process an Information Resource Directory to identify the URI corresponding to a desired Information Resource. However, some advice is provided for implementors.

It is possible that multiple entries in the directory match a desired

Information Resource. For instance, in the example in Section 7.6.3, a full Cost Map with "numerical" Cost Mode and "routingcost" Cost Type could be retrieved via a GET request to "http://alto.example.com/costmap/num/routingcost", or via a POST request to "http://custom.alto.example.com/costmap/filtered".

In general, it is preferred for ALTO Clients to use GET requests where appropriate, since it is more likely for responses to be cacheable.

7.6.4.2. ALTO Server

This document indicates that an ALTO Server may or may not provide the Information Resources specified in the Map Filtering Service. If these resources are not provided, it is indicated to an ALTO Client by the absence of a Network Map or Cost Map with any media types listed under "accepts".

7.7. Information Resources

This section documents the individual Information Resources defined in the ALTO Protocol.

7.7.1. Server Information Service

The Server Information Service provides generic information about an ALTO Server.

7.7.1.1. Server Info

This Information Resource MUST be provided by an ALTO Server.

7.7.1.1.1. Media Type

The media type is "application/alto-serverinfo+json".

7.7.1.1.2. HTTP Method

This resource is requested using the HTTP GET method.

7.7.1.1.3. Input Parameters

None.

7.7.1.1.4. Capabilities

None.

7.7.1.1.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceServerInfo:

```
object {
  JSONString   service-id;            [OPTIONAL]
  JSONString   certificates<0..*>;    [OPTIONAL]
} InfoResourceServerInfo;
```

which has members:

service-id UUID [RFC4122] indicating an one or more ALTO Servers serving equivalent ALTO Information.

certificates List of PEM-encoded X.509 certificates used by the ALTO Server in the signing of responses.

If an ALTO Server has the possibility of marking any response as redistributable, the 'service-id' and 'certificates' fields are REQUIRED instead of OPTIONAL. See Section 8&$160;for detailed specification.

7.7.1.1.6. Example

```
GET /serverinfo HTTP/1.1
Host: alto.example.com
Accept: application/alto-serverinfo+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-serverinfo+json

{
  "meta" : {},
  "data" : {
    "service-id" : "c89ca72f-dead-41b5-9e2b-b65455ace1ee",
    "certificates" : [ ... ]
  }
}
```

7.7.2. Map Service

The Map Service provides batch information to ALTO Clients in the form of two types of maps: a Network Map and Cost Map.

7.7.2.1. Network Map

The Network Map Information Resource lists for each PID, the network locations (endpoints) within the PID. It MUST be provided by an ALTO Server.

7.7.2.1.1. Media Type

The media type is "application/alto-networkmap+json".

7.7.2.1.2. HTTP Method

This resource is requested using the HTTP GET method.

7.7.2.1.3. Input Parameters

None.

7.7.2.1.4. Capabilities

None.

7.7.2.1.5. Response

The returned InfoResourceEntity object "data" member of type InfoResourceNetworkMap:

```
object {
  EndpointAddrGroup [pidname]<0..*>;
  ...
} NetworkMapData;

object {
  JSONString     map-vtag;
  NetworkMapData map;
} InfoResourceNetworkMap;
```

with members:

map-vtag The Version Tag (Section 5.3) of the Network Map.

map The Network Map data itself.

NetworkMapData is a JSON object with each member representing a single PID and its associated set of endpoint addresses. A member's name is a string of type PIDName.

The returned Network Map MUST include all PIDs known to the ALTO Server.

7.7.2.1.6. Example

```
GET /networkmap HTTP/1.1
Host: alto.example.com
Accept: application/alto-networkmap+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-networkmap+json

{
  "meta" : {},
  "data" : {
    "map-vtag" : "1266506139",
    "map" : {
      "PID1" : {
        "ipv4" : [
          "192.0.2.0/24",
          "198.51.100.0/25"
        ]
      },
      "PID2" : {
        "ipv4" : [
          "198.51.100.128/25"
        ]
      },
      "PID3" : {
        "ipv4" : [
          "0.0.0.0/0"
        ],
        "ipv6" : [
          "::/0"
        ]
      }
    }
  }
}
```

7.7.2.2. Cost Map

The Cost Map resource lists the Path Cost for each pair of source/ destination PID defined by the ALTO Server for a given Cost Type and Cost Mode. This resource MUST be provided for at least the 'routingcost' Cost Type and 'numerical' Cost Mode.

Note that since this resource, an unfiltered Cost Map requested by an HTTP GET, does not indicate the desired Cost Mode or Cost Type as input parameters, an ALTO Server MUST indicate in an Information Resource Directory a unfiltered Cost Map Information Resource by specifying the capabilities (Section 7.7.2.2.4) with "cost-types" and "cost-modes" members each having a single element. This technique will allow an ALTO Client to determine a URI for an unfiltered Cost Map of the desired Cost Mode and Cost Type.

7.7.2.2.1. Media Type

The media type is "application/alto-costmap+json".

7.7.2.2.2. HTTP Method

This resource is requested using the HTTP GET method.

7.7.2.2.3. Input Parameters

None.

7.7.2.2.4. Capabilities

This resource may be defined for across multiple Cost Types and Cost Modes. The capabilities of an ALTO Server URI providing this resource are defined by a JSON Object of type CostMapCapability:

```
object {
  CostMode cost-modes<0..*>;
  CostType cost-types<0..*>;
} CostMapCapability;
```

with members:

cost-modes The Cost Modes (Section 5.1.2) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

cost-types The Cost Types (Section 5.1.1) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

An ALTO Server MUST support all of the Cost Types listed here for each of the listed Cost Modes. Note that an ALTO Server may provide multiple Cost Map Information Resources, each with different capabilities.

7.7.2.2.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceCostMap:

```
object DstCosts {
  JSONNumber [PIDName];
  ...
};

object {
  DstCosts [PIDName]<0..*>;
  ...
} CostMapData;

object {
  CostMode    cost-mode;
  CostType    cost-type;
  JSONString  map-vtag;
  CostMapData map;
} InfoResourceCostMap;
```

with members:

cost-mode Cost Mode (Section 5.1.2) used in the Cost Map.

cost-type Cost Type (Section 5.1.1) used in the Cost Map.

map-vtag The Version Tag (Section 5.3) of the Network Map used to generate the Cost Map.

map The Cost Map data itself.

CostMapData is a JSON object with each member representing a single Source PID; the name for a member is the PIDName string identifying the corresponding Source PID. For each Source PID, a DstCosts object denotes the associated cost to a set of destination PIDs (Section 5.2); the name for each member in the object is the PIDName string identifying the corresponding Destination PID. DstCosts MUST have a single member for each Destination PID in the map.

The returned Cost Map MUST include Path Costs for each pair of Source PID and Destination PID known to the ALTO Server.

7.7.2.2.6. Example

```
GET /costmap/num/routingcost HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : "numerical",
    "cost-type" : "routingcost",
    "map-vtag"  : "1266506139",
    "map" : {
      "PID1": { "PID1": 1,  "PID2": 5,  "PID3": 10 },
      "PID2": { "PID1": 5,  "PID2": 1,  "PID3": 15 },
      "PID3": { "PID1": 20, "PID2": 15, "PID3": 1  }
    }
  }
}
```

7.7.3. Map Filtering Service

The Map Filtering Service allows ALTO Clients to specify filtering criteria to return a subset of the full maps available in the Map Service.

7.7.3.1. Filtered Network Map

A Filtered Network Map is a Network Map Information Resource (Section 7.7.2.1) for which an ALTO Client may supply a list of PIDs to be included. A Filtered Network Map MAY be provided by an ALTO Server.

7.7.3.1.1. Media Type

See Section 7.7.2.1.1.

7.7.3.1.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.3.1.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-networkmapfilter+json", which is a JSON Object of type ReqFilteredNetworkMap, where:

```
object {
  PIDName pids<0..*>;
} ReqFilteredNetworkMap;
```

with members:

pids Specifies list of PIDs to be included in the returned Filtered Network Map. If the list of PIDs is empty, the ALTO Server MUST interpret the list as if it contained a list of all currently-defined PIDs. The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

7.7.3.1.4. Capabilities

None.

7.7.3.1.5. Response

See Section 7.7.2.1.5 for the format.

The ALTO Server MUST only include PIDs in the response that were specified (implicitly or explicitly) in the request. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as if the PID did not appear in the input parameters.

7.7.3.1.6. Example

```
POST /networkmap/filtered HTTP/1.1
Host: custom.alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-networkmapfilter+json
Accept: application/alto-networkmap+json,application/alto-error+json

{
  "pids": [ "PID1", "PID2" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-networkmap+json

{
  "meta" : {},
  "data" : {
    "map-vtag" : "1266506139",
    "map" : {
      "PID1" : {
        "ipv4" : [
          "192.0.2.0/24",
          "198.51.100.0/24"
        ]
      },
      "PID2" : {
        "ipv4": [
          "198.51.100.128/24"
        ]
      }
    }
  }
}
```

7.7.3.2. Filtered Cost Map

A Filtered Cost Map is a Cost Map Information Resource (Section 7.7.2.2) for which an ALTO Client may supply additional parameters limiting the scope of the resulting Cost Map. A Filtered Cost Map MAY be provided by an ALTO Server.

7.7.3.2.1. Media Type

See Section 7.7.2.2.1.

7.7.3.2.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.3.2.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-costmapfilter+json", which is a JSON Object of type ReqFilteredCostMap, where:

```
object {
  PIDName srcs<0..*>;
  PIDName dsts<0..*>;
} PIDFilter;

object {
  CostMode   cost-mode;
  CostType   cost-type;
  JSONString constraints<0..*>;   [OPTIONAL]
  PIDFilter  pids;                [OPTIONAL]
} ReqFilteredCostMap;
```

with members:

cost-type The Cost Type (Section 5.1.1) for the returned costs. This MUST be one of the supported Cost Types indicated in this resource's capabilities (Section 7.7.3.2.4).

cost-mode The Cost Mode (Section 5.1.2) for the returned costs. This MUST be one of the supported Cost Modes indicated in this resource's capabilities (Section 7.7.3.2.4).

constraints Defines a list of additional constraints on which elements of the Cost Map are returned. This parameter MUST NOT be specified if this resource's capabilities (Section 7.7.3.2.4) indicate that constraint support is not available. A constraint contains two entities separated by whitespace: (1) an operator either 'gt' for greater than or 'lt' for less than (2) a target numerical cost. The numerical cost is a number that MUST be defined in the same units as the Cost Type indicated by the cost-type parameter. ALTO Servers SHOULD use at least IEEE 754 double-

precision floating point [IEEE.754.2008] to store the numerical cost, and SHOULD perform internal computations using double-precision floating-point arithmetic. If multiple 'constraint' parameters are specified, they are interpreted as being related to each other with a logical AND.

pids A list of Source PIDs and a list of Destination PIDs for which Path Costs are to be returned. If a list is empty, the ALTO Server MUST interpret it as the full set of currently-defined PIDs. The ALTO Server MUST interpret entries appearing in a list multiple times as if they appeared only once. If the "pids" member is not present, both lists MUST be interpreted by the ALTO Server as containing the full set of currently-defined PIDs.

7.7.3.2.4. Capabilities

The URI providing this resource supports all capabilities documented in Section 7.7.2.2.4 (with identical semantics), plus additional capabilities. In particular, the capabilities are defined by a JSON object of type FilteredCostMapCapability:

```
object {
  CostMode cost-modes<0..*>;
  CostType cost-types<0..*>;
  JSONBool cost-constraints;
} FilteredCostMapCapability;
```

with members:

cost-modes See Section 7.7.2.2.4.

cost-types See Section 7.7.2.2.4.

cost-constraints If true, then the ALTO Server allows cost constraints to be included in requests to the corresponding URI. If not present, this member MUST be interpreted as if it specified false.

7.7.3.2.5. Response

See Section 7.7.2.2.5 for the format.

The returned Cost Map MUST NOT contain any source/destination pair that was not indicated (implicitly or explicitly) in the input parameters. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as

if the PID did not appear in the input parameters.

If any constraints are specified, Source/Destination pairs that do for which the Path Costs do not meet the constraints MUST NOT be included in the returned Cost Map. If no constraints were specified, then all Path Costs are assumed to meet the constraints.

7.7.3.2.6. Example

```
POST /costmap/filtered HTTP/1.1
Host: custom.alto.example.com
Content-Type: application/alto-costmapfilter+json
Accept: application/alto-costmap+json,application/alto-error+json

{
  "cost-mode" : "numerical",
  "cost-type" : "routingcost",
  "pids" : {
    "srcs" : [ "PID1" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : "numerical",
    "cost-type" : "routingcost",
    "map-vtag" : "1266506139",
    "map" : {
      "PID1": { "PID1": 0,  "PID2": 1,  "PID3": 2 }
    }
  }
}
```

7.7.4. Endpoint Property Service

The Endpoint Property Service provides information about Endpoint properties to ALTO Clients.

7.7.4.1. Endpoint Property

The Endpoint Property resource provides information about properties for individual endpoints. It MAY be provided by an ALTO Server. If an ALTO Server provides the Endpoint Property resource, it MUST provide and define at least the 'pid' property for each Endpoint.

7.7.4.1.1. Media Type

The media type is "application/alto-endpointprop+json".

7.7.4.1.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.4.1.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the data format of input parameteres with the media type "application/alto-endpointpropparams+json", which is a JSON Object of type ReqEndpointProp:

```
object {
  EndpointProperty  properties<1..*>;
  TypedEndpointAddr endpoints<1..*>;
} ReqEndpointProp;
```

with members:

properties List of endpoint properties to returned for each endpoint. Each specified property MUST be included in the list of supported properties indicated by this resource's capabilities (Section 7.7.4.1.4). The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

endpoints List of endpoint addresses for which the specified properties are to be returned. The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

7.7.4.1.4. Capabilities

This resource may be defined across multiple types of endpoint properties. The capabilities of an ALTO Server URI providing Endpoint Properties are defined by a JSON Object of type EndpointPropertyCapability:

```
object {
  EndpointProperty prop-types<0..*>;
} EndpointPropertyCapability;
```

with members:

prop-types The Endpoint Property Types (Section 3.2.3) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

7.7.4.1.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceEndpointProperty, where:

```
object {
  JSONString [EndpointProperty];
  ...
} EndpointProps;

object {
  EndpointProps  [TypedEndpointAddr]<0..*>;
  ...
} InfoResourceEndpointProperty;
```

InfoResourceEndpointProperty has one member for each endpoint indicated in the input parameters (with the name being the endpoint encoded as a TypedEndpointAddr). The requested properties for each endpoint are encoded in a corresponding EndpointProps object, which encodes one name/value pair for each requested property, where the property names are encoded as strings of type EndpointProperty and the property values encoded as JSON Strings.

The ALTO Server returns the value for each of the requested endpoint properties for each of the endpoints listed in the input parameters.

If the ALTO Server does not define a requested property for a particular endpoint, then it MUST omit it from the response for only that endpoint.

7.7.4.1.6. Example

```
POST /endpointprop/lookup HTTP/1.1
Host: alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-endpointpropparams+json
Accept: application/alto-endpointprop+json,application/alto-error+json

{
  "properties" : [ "pid" ],
  "endpoints" : [ "ipv4:192.0.2.34", "ipv4:203.0.113.129" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-endpointprop+json

{
  "meta" : {},
  "data": {
    "ipv4:192.0.2.34"    : { "pid": "PID1" },
    "ipv4:203.0.113.129" : { "pid": "PID3" }
  }
}
```

7.7.5. Endpoint Cost Service

The Endpoint Cost Service provides information about costs between individual endpoints.

In particular, this service allows lists of Endpoint prefixes (and addresses, as a special case) to be ranked (ordered) by an ALTO Server.

7.7.5.1. Endpoint Cost

The Endpoint Cost resource provides information about costs between individual endpoints. It MAY be provided by an ALTO Server. If it is provided.

It is important to note that although this resource allows an ALTO Server to reveal costs between individual endpoints, an ALTO Server is not required to do so. A simple alternative would be to compute the cost between two endpoints as the cost between the PIDs corresponding to the endpoints. See Section 12.1 for additional

details.

7.7.5.1.1. Media Type

The media type is "application/alto-endpointcost+json".

7.7.5.1.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.5.1.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies input parameters with a data format indicated by media type "application/alto-endpointcostparams+json", which is a JSON Object of type ReqEndpointCostMap:

```
object {
  TypedEndpointAddr srcs<0..*>;    [OPTIONAL]
  TypedEndpointAddr dsts<1..*>;
} EndpointFilter;

object {
  CostMode          cost-mode;
  CostType          cost-type;
  JSONString        constraints;   [OPTIONAL]
  EndpointFilter    endpoints;
} ReqEndpointCostMap;
```

with members:

cost-mode The Cost Mode (Section 5.1.2) to use for returned costs. This MUST be one of the Cost Modes indicated in this resource's capabilities (Section 7.7.5.1.4).

cost-type The Cost Type (Section 5.1.1) to use for returned costs. This MUST be one of the Cost Types indicated in this resource's capabilities (Section 7.7.5.1.4).

constraints Defined equivalently to the "constraints" input parameter of a Filtered Cost Map (see Section 7.7.3.2).

endpoints A list of Source Endpoints and Destination Endpoints for which Path Costs are to be returned. If the list of Source Endpoints is empty (or not included), the ALTO Server MUST interpret it as if it contained the Endpoint Address corresponding

to the client IP address from the incoming connection (see Section 10.3 for discussion and considerations regarding this mode). The list of destination Endpoints MUST NOT be empty. The ALTO Server MUST interpret entries appearing multiple times in a list as if they appeared only once.

7.7.5.1.4. Capabilities

See Section 7.7.3.2.4.

7.7.5.1.5. Response

The returned InfoResourceEntity object has "data" member equal to InfoResourceEndpointCostMap, where:

```
object EndpointDstCosts {
  JSONNumber [TypedEndpointAddr];
  ...
};

object {
  EndpointDstCosts [TypedEndpointAddr]<0..*>;
  ...
} EndpointCostMapData;

object {
  CostMode            cost-mode;
  CostType            cost-type;
  EndpointCostMapData map;
} InfoResourceEndpointCostMap;
```

InfoResourceEndpointCostMap has members:

cost-mode The Cost Mode used in the returned Cost Map.

cost-type The Cost Type used in the returned Cost Map.

map The Endpoint Cost Map data itself.

EndpointCostMapData is a JSON object with each member representing a single Source Endpoint specified in the input parameters; the name for a member is the TypedEndpointAddr string identifying the corresponding Source Endpoint. For each Source Endpoint, a EndpointDstCosts object denotes the associated cost to each Destination Endpoint specified in the input parameters; the name for each member in the object is the TypedEndpointAddr string identifying

the corresponding Destination Endpoint.

7.7.5.1.6. Example

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-endpointcostparams+json
Accept: application/alto-endpointcost+json,application/alto-error+json

{
  "cost-mode" : "ordinal",
  "cost-type" : "routingcost",
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34",
      "ipv4:203.0.113.45"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-endpointcost+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : "ordinal",
    "cost-type" : "routingcost",
    "map" : {
      "ipv4:192.0.2.2": {
        "ipv4:192.0.2.89"    : 1,
        "ipv4:198.51.100.34" : 2,
        "ipv4:203.0.113.45"  : 3
      }
    }
  }
}
```

8. Redistributable Responses

This section defines how an ALTO Server enables certain Information Resources to be redistributed by ALTO Clients. Concepts are first introduced, followed by the protocol specification.

8.1. Concepts

8.1.1. Service ID

The Service ID is a UUID that identifies a set of ALTO Servers that would provide semantically-identical Information Resources for any request for any ALTO Client. Each ALTO Server within such a set is configured with an identical Service ID.

If a pair of ALTO Servers would provide an identical Information Resource (same information sources, configuration, internal computations, update timescales, etc) in response to any particular ALTO Client request, then the pair of ALTO Servers MAY have the same Service ID. If this condition is not true, the pair of ALTO Servers MUST have a different Service ID.

8.1.1.1. Rationale

For scalability and fault tolerance, multiple ALTO Servers may be deployed to serve equivalent ALTO Information. In such a scenario, Information Resources from any such redundant server should be seen as equivalent for the purposes of redistribution. For example, if two ALTO Servers A and B are deployed by the service provider to distribute equivalent ALTO Information, then clients contacting Server A should be able to redistribute Information Resources to clients contacting Server B.

To accomplish this behavior, ALTO Clients must be able to determine that Server A and Server B serve identical ALTO Information. One technique would be to rely on the ALTO Server's DNS name. However, such an approach would mandate that all ALTO Servers resolved by a particular DNS name would need to provide equivalent ALTO information, which may be unnecessarily restrictive. Another technique would be to rely on the server's IP address. However, this suffers similar problems as the DNS name in deployment scenarios using IP Anycast.

To avoid such restrictions, the ALTO Protocol allows an ALTO Service Provider to explicitly denote ALTO Servers that provide equivalent ALTO Information by giving them identical Service IDs. Service IDs decouple the identification of equivalent ALTO Servers from the discovery process.

8.1.1.2. Server Information Resource

If an ALTO Server generates redistributable responses, the Server Information resource's 'service-id' field MUST be set to the ALTO Server's Service ID.

8.1.1.3. Configuration

To help prevent ALTO Servers from mistakenly claiming to distribute equivalent ALTO Information, ALTO Server implementations SHOULD by default generate a new UUID at installation time or startup if one has not explicitly been configured.

8.1.2. Expiration Time

Information Resources marked as redistributable should indicate a time after which the information is considered stale and should be refreshed from the ALTO Server (or possibly another ALTO Client).

If an expiration time is present, the ALTO Server SHOULD ensure that it is reasonably consistent with the expiration time that would be computed by HTTP header fields. This specification makes no recommendation on which expiration time takes precedence, but implementers should be cognizant that HTTP intermediaries will obey only the HTTP header fields.

8.1.3. Signature

Information Resources marked as redistributable include a signature used to assert that the ALTO Server Provider generated the ALTO Information.

8.1.3.1. Rationale

Verification of the signature requires the ALTO Client to retrieve the ALTO Server's public key. To reduce requirements on the underlying transport (i.e., requiring SSL/TLS), an ALTO Client retrieves the public key as part of an X.509 certificate from the ALTO Server's Server Information resource.

8.1.3.2. Certificates

8.1.3.2.1. Local Certificate

The ALTO Server's public key is encoded within an X.509 certificate. The corresponding private key MUST be used to sign redistributable responses. This certificate is termed the Local Certificate for an ALTO Server.

8.1.3.2.2. Certificate Chain

To ease key provisioning, the ALTO Protocol is designed such that each ALTO Server with an identical Service ID may have a unique private key (and hence certificate).

The ALTO Service Provider may configure a certificate chain at each such ALTO Server. The Local Certificate for a single ALTO Server is the bottom-most certificate in the chain. The Certificate Chains of each ALTO Server with an identical Service ID MUST share a common Root Certificate.

Note that there are two simple deployment scenarios:

- One-Level Certificate Chain (Local Certificate Only): In this deployment scenario, each ALTO Server with an identical Service ID may provisioned with an identical Local Certificate.

- Two-Level Certificate Chain: In this deployment scenario, a Root Certificate is maintained for a set of ALTO Servers with the same Service ID. A unique Local Certificate signed by this CA is provisioned to each ALTO Server.

There are advantages to using a Certificate Chain instead of deploying the same Local Certificate to each ALTO Server. Specifically, it avoids storage of the CA's private key at ALTO Servers. It is possible to revoke and re-issue a key to a single ALTO Server.

8.1.3.2.3. Server Information Resource

If an ALTO Server generates redistributable responses, the Server Information resource's 'certificates' field MUST be populated with the ALTO Server's full certificate chain. The first element MUST be the ALTO Server's Local Certificate, followed by the remaining Certificate Chain in ascending order to the Root Certificate.

8.1.3.3. Signature Verification

ALTO Clients SHOULD verify the signature on any ALTO information received via redistribution before adjusting application behavior based on it.

An ALTO Client SHOULD cache its ALTO Server's Service ID and corresponding Certificate Chain included in the Server Information resource. Recall that the last certificate in this chain is the Root Certificate. The retrieval of the Service ID and certificates SHOULD be secured using HTTPS with proper validation of the server endpoint

of the SSL/TLS connection [RFC6125].

An Information Resource received via redistribution from Service ID S is declared valid if an ALTO Client can construct a transitive certificate chain from the certificate (public key) used to sign the Information Resource to the Root Certificate corresponding to Service ID S obtained by the ALTO Client in a Server Information resource.

To properly construct the chain and complete this validation, an ALTO Client may need to request additional certificates from other ALTO Clients. A simple mechanism is to request the certificate chain from the ALTO Client that received the Information Resource. Note that these additional received certificates may be cached locally by an ALTO Client.

ALTO Clients SHOULD verify Information Resources received via redistribution.

8.1.3.4. Redistribution by ALTO Clients

ALTO Clients SHOULD pass the ALTO Server Certificate, Signature, and Signature Algorithm along with the Information Resource. The mechanism for redistributing such information is not specified by the ALTO Protocol, but one possibility is to add additional messages or fields to the application's native protocol.

8.2. Protocol

An ALTO Server MAY indicate that a response is suitable for redistribution by including the "redistribution" member in the RspMetaData JSON object of an Information Resource. This additional member, called the Response Redistribution Descriptor, has type InfoResourceRedistDesc:

```
object {
  JSONString service-id;
  JSONString request-uri;
  JSONValue  request-body;
  JSONString media-type;
  JSONString expires;
} InfoResourceRedistDesc;
```

The fields encoded in the Response Redistribution Descriptor allows an ALTO Client receiving redistributed ALTO Information to understand the context of the query (the ALTO Service generating the response and any input parameters) and to interpret the results.

Information about ALTO Client performing the request and any HTTP

Headers passed in the request are not included in the Response Redistribution Descriptor. If any such information or headers influence the response generated by the ALTO Server, the response SHOULD NOT be indicated as redistributable.

8.2.1. Response Redistribution Descriptor Fields

This section defines the fields of the Response Redistribution Descriptor.

8.2.1.1. Service ID

The 'service-id' member is REQUIRED and MUST have a value equal to the ALTO Server's Service ID.

8.2.1.2. Request URI

The 'request-uri' member is REQUIRED and MUST specify the HTTP Request-URI that was passed in the HTTP request.

8.2.1.3. Request Body

If the HTTP request's entity body was non-empty, the 'request-body' member MUST specify full JSON value passed in the HTTP request's entity body (note that whitespace may differ, as long as the JSON Value is identical). If the HTTP request was empty, then the 'request-body' MUST NOT be included.

8.2.1.4. Response Media Type

The 'media-type' member is REQUIRED and MUST specify the same HTTP Content-Type that is used in the HTTP response.

8.2.1.5. Expiration Time

The 'expires' element is RECOMMENDED and, if present, MUST specify a time in UTC formatted according to [RFC3339].

8.2.2. Signature

The Hash Algorithm, Signature Algorithm, and Signature are included as either HTTP Headers or Trailers. Headers may be useful if Information Resources are pre-generated, while Trailers may be useful if Information Resources are dynamically generated (e.g., to avoid buffering large responses in memory while the hash value is computed).

The following HTTP Headers (the ALTO Server MAY specify them as HTTP

Trailers instead) MUST be used to encode the Signature parameters for redistributable Information Resources:

```
ALTO-HashAlgorithm: <HashAlgorithm>
ALTO-SignatureAlgorithm: <SignatureAlgorithm>
ALTO-SignatureDigest: <Signature>
```

where <HashAlgorithm> and <SignatureAlgorithm> are an integer values from the IANA TLS HashAlgorithm and SignatureAlgorithm registries, and <Signature> is the corresponding Base64-encoded signature.

9. Use Cases

The sections below depict typical use cases.

9.1. 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.

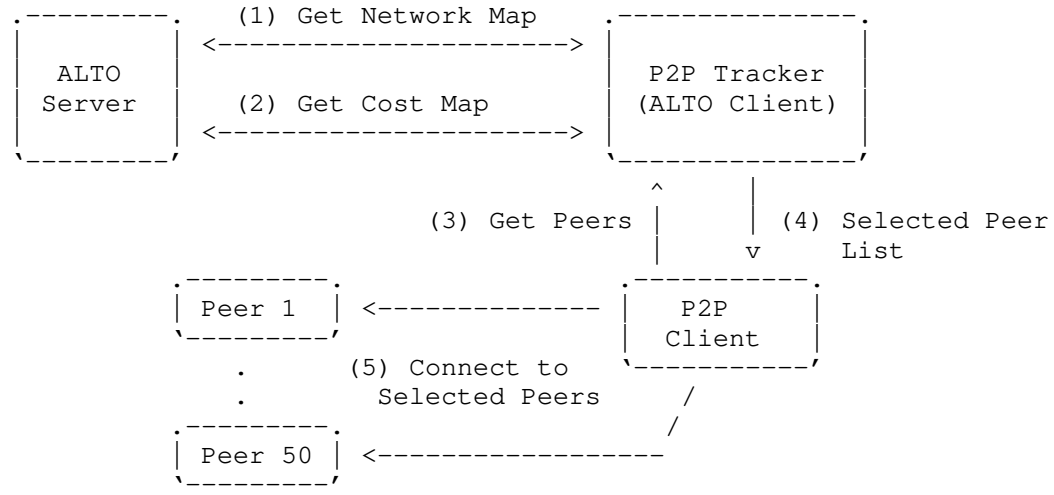


Figure 4: ALTO Client Embedded in P2P Tracker

Figure 4 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 Network Map 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.

9.2. 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.

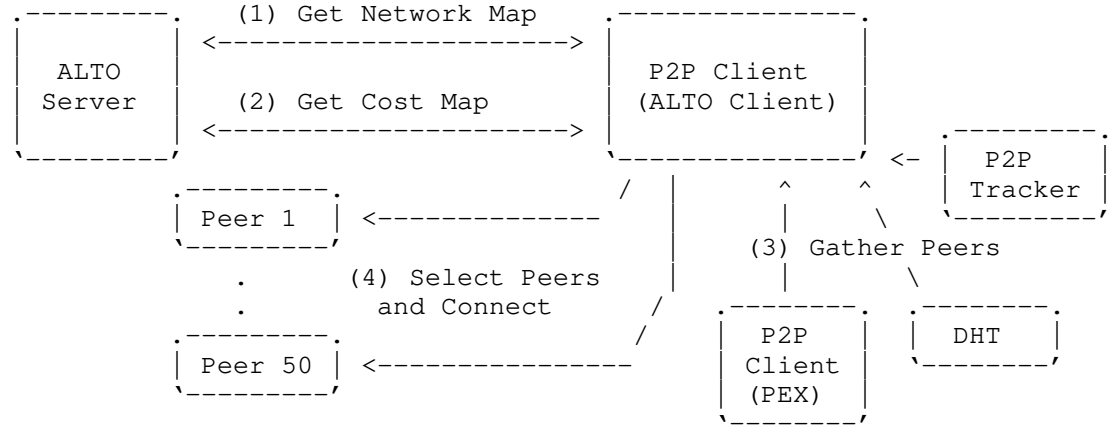


Figure 5: ALTO Client Embedded in P2P Client

Figure 5 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.

9.3. 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.

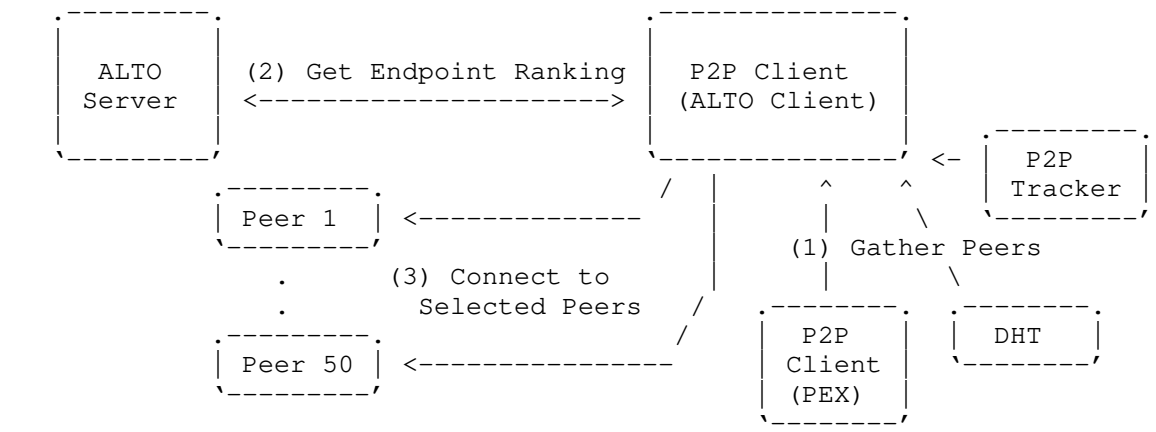


Figure 6: ALTO Client Embedded in P2P Client: Ranking

Figure 6 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 the ALTO Server's Ranking Service, including discovered peers as the set of Destination Endpoints, and indicates the 'ordinal' Cost Mode. The response indicates the ranking of the candidate peers.

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

10. Discussions

10.1. Discovery

The discovery mechanism by which an ALTO Client locates an appropriate ALTO Server is out of scope for this document. This document assumes that an ALTO Client can discover an appropriate ALTO Server. Once it has done so, the ALTO Client may use the Information Resource Directory (see Section 7.6) to locate an Information Resource with the desired ALTO Information.

10.2. Hosts with Multiple Endpoint Addresses

In practical deployments, especially during the transition from IPv4 to IPv6, a particular host may be reachable using multiple addresses. Furthermore, the particular network path followed when sending packets to the host may differ based on the address that is used. Network providers may prefer one path over another (e.g., one path my have a NAT64 middlebox). An additional consideration may be how to handle private address spaces (e.g., behind carrier-grade NATs).

To support such behavior, this document allows multiple types of endpoint addresses. In supporting multiple address types, the ALTO Protocol also allows ALTO Service Provider the flexibility to indicate preferences for paths from an endpoint address of one type to an endpoint address of a different type. Note that in general, the path through the network may differ dependent on the types of addresses that are used.

Note that there are limitations as to what information ALTO can provide in this regard. In particular, a particular ALTO Service provider may not be able to determine if connectivity with a particular endhost will succeed over IPv4 or IPv6, as this may depend upon information unknown to the ISP such as particular application implementations.

10.3. Network Address Translation Considerations

At this day and age of NAT v4<->v4, v4<->v6 [RFC6144], and possibly v6<->v6[I-D.mrw-nat66], 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 where the source network location is computed by the ALTO Server (i.e., the the Endpoint Cost Service) 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 [BitTorrent]) operate.

The ALTO client SHOULD use the Session Traversal Utilities for NAT (STUN) [RFC5389] to determine a public IP address to use as a source Endpoint address. If using this method, the host MUST use 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.

10.4. 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, the Network Map query in the Map Filtering Service 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.

10.5. 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. This specification focuses on network location and grouping, but the protocol may be extended to handle other Endpoint properties.

11. IANA Considerations

11.1. application/alto-* Media Types

This document requests the registration of multiple media types, listed in Table 2.

Type	Subtype	Specification
application	alto-directory+json	Section 7.6
application	alto-serverinfo+json	Section 7.7.1.1
application	alto-networkmap+json	Section 7.7.2.1
application	alto-networkmapfilter+json	Section 7.7.3.1
application	alto-costmap+json	Section 7.7.2.2
application	alto-costmapfilter+json	Section 7.7.3.2
application	alto-endpointprop+json	Section 7.7.4.1
application	alto-endpointpropparams+json	Section 7.7.4.1
application	alto-endpointcost+json	Section 7.7.5.1
application	alto-endpointcostparams+json	Section 7.7.5.1
application	alto-error+json	Section 7.4

Table 2: ALTO Protocol Media Types

Type name: application

Subtype name: This documents requests the registration of multiple subtypes, as listed in Table 2.

Required parameters: n/a

Optional parameters: n/a

Encoding considerations: Encoding considerations are identical to those specified for the 'application/json' media type. See [RFC4627].

Security considerations: Security considerations relating to the generation and consumption of ALTO protocol messages are discussed in Section 12.

Interoperability considerations: This document specifies format of conforming messages and the interpretation thereof.

Published specification: This document is the specification for these media types; see Table 2for the section documenting each media type.

Applications that use this media type: ALTO Servers and ALTO Clients either standalone or embedded within other applications.

Additional information:

Magic number(s): n/a

File extension(s): This document uses the mime type to refer to protocol messages and thus does not require a file extension.

Macintosh file type code(s): n/a

Person & email address to contact for further information: See "Authors' Addresses" section.

Intended usage: COMMON

Restrictions on usage: n/a

Author: See "Authors' Addresses" section.

Change controller: See "Authors' Addresses" section.

11.2. ALTO Cost Type Registry

This document requests the creation of an ALTO Cost Type registry to be maintained by IANA.

This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO Cost Types. Second, it provides references to particular semantics of allocated Cost Types to be applied by both ALTO Servers and applications utilizing ALTO Clients.

New ALTO Cost Types are assigned after Expert Review [RFC5226]. The Expert Reviewer will generally consult the ALTO Working Group or its successor. Expert Review is used to ensure that proper documentation regarding ALTO Cost Type semantics and security considerations has been provided. The provided documentation should be detailed enough to provide guidance to both ALTO Service Providers and applications utilizing ALTO Clients as to how values of the registered ALTO Cost Type should be interpreted. Updates and deletions of ALTO Cost Types follow the same procedure.

Registered ALTO Cost Type identifiers MUST conform to the syntatical requirements specified in Section 7.5.4. Identifiers are to be recorded and displayed as ASCII strings.

Identifiers prefixed with 'priv:' are reserved for Private Use. Identifiers prefixed with 'exp:' are reserved for Experimental use.

Requests to add a new value to the registry MUST include the

following information:

- Identifier: The name of the desired ALTO Cost Type.

- Intended Semantics: ALTO Costs carry with them semantics to guide their usage by ALTO Clients. For example, if a value refers to a measurement, the measurement units must be documented. For proper implementation of the ordinal Cost Mode (e.g., by a third-party service), it should be documented whether higher or lower values of the cost are more preferred.

- Security Considerations: ALTO Costs expose information to ALTO Clients. As such, proper usage of a particular Cost Type may require certain information to be exposed by an ALTO Service Provider. Since network information is frequently regarded as proprietary or confidential, ALTO Service Providers should be made aware of the security ramifications related to usage of a Cost Type.

This specification requests registration of the identifier 'routingcost'. Semantics for the this Cost Type are documented in Section 5.1.1.1, and security considerations are documented in Section 12.1.

11.3. ALTO Endpoint Property Registry

This document requests the creation of an ALTO Endpoint Property registry to be maintained by IANA.

This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO Endpoint Properties. Second, it provides references to particular semantics of allocated Endpoint Properties to be applied by both ALTO Servers and applications utilizing ALTO Clients.

New ALTO Endpoint Properties are assigned after Expert Review [RFC5226]. The Expert Reviewer will generally consult the ALTO Working Group or its successor. Expert Review is used to ensure that proper documentation regarding ALTO Endpoint Property semantics and security considerations has been provided. The provided documentation should be detailed enough to provide guidance to both ALTO Service Providers and applications utilizing ALTO Clients as to how values of the registered ALTO Endpoint Properties should be interpreted. Updates and deletions of ALTO Endpoint Properties follow the same procedure.

Registered ALTO Endpoint Property identifiers MUST conform to the syntatical requirements specified in Section 7.5.5. Identifiers are

to be recorded and displayed as ASCII strings.

Identifiers prefixed with 'priv:' are reserved for Private Use. Identifiers prefixed with 'exp:' are reserved for Experimental use.

Requests to add a new value to the registry MUST include the following information:

- Identifier: The name of the desired ALTO Endpoint Property.
- Intended Semantics: ALTO Endpoint Properties carry with them semantics to guide their usage by ALTO Clients. For example, if a value refers to a measurement, the measurement units must be documented. For proper implementation of the ordinal Cost Mode (e.g., by a third-party service), it should be documented whether higher or lower values of the cost are more preferred.
- Security Considerations: ALTO Endpoint Properties expose information to ALTO Clients. As such, proper usage of a particular Endpoint Properties may require certain information to be exposed by an ALTO Service Provider. Since network information is frequently regarded as proprietary or confidential, ALTO Service Providers should be made aware of the security ramifications related to usage of an Endpoint Property.

This specification requests registration of the identifier 'pid'. Semantics for the this Endpoint Property are documented in Section 4.1, and security considerations are documented in Section 12.1.

12. Security Considerations

12.1. Privacy Considerations for 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. On the one hand, providing overly fine-grained information may make it easier for attackers to infer network topology. In particular, attackers may try to infer details regarding ISPs' operational policies or inter-ISP business relationships by intentionally posting a multitude of selective queries to an ALTO server and analyzing the responses. Such sophisticated attacks may reveal more information than an ISP hosting an ALTO server intends to disclose. On the other hand, revealing overly coarse-grained information may not provide benefits to network efficiency or performance improvements to ALTO Clients.

12.2. ALTO Clients

Applications using the information must be cognizant of the possibility that the information is malformed or incorrect. Even if an ALTO Server has been properly authenticated by the ALTO Client, the information provided may be malicious because the ALTO Server and its credentials have been compromised (e.g., through malware). Other considerations (e.g., relating to application performance) can be found in Section 6 of [RFC5693].

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.

In addition, ALTO clients should be cautious not to unintentionally or indirectly disclose the resource identifier (of which they try to improve the retrieval through ALTO-guidance), e.g., the name/ identifier of a certain video stream in P2P live streaming, to the ALTO server. Note that the ALTO Protocol specified in this document does not explicitly reveal any resource identifier to the ALTO Server. However, for instance, depending on the popularity or other specifics (such as language) of the resource, an ALTO server could potentially deduce information about the desired resource from information such as the Network Locations the client sends as part of its request to the server.

12.3. Authentication, Integrity Protection, and Encryption

SSL/TLS can provide encryption of transmitted messages as well as authentication of the ALTO Client and Server. HTTP Basic or Digest authentication can provide authentication of the client (combined with SSL/TLS, it can additionally provide encryption and authentication of the server).

An ALTO Server may optionally use authentication (and potentially encryption) to protect ALTO information it provides. This can be achieved by digitally signing a hash of the ALTO information itself and attaching the signature to the ALTO information. There may be special use cases where encryption of ALTO information is desirable. In many cases, however, information sent out by an ALTO Server may be regarded as non-confidential information.

ISPs should be cognizant that encryption only protects ALTO information until it is decrypted by the intended ALTO Client. Digital Rights Management (DRM) techniques and legal agreements

protecting ALTO information are outside of the scope of this document.

12.4. 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 generated by an appropriate 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 validate every set of information. Contacting an ALTO server for information validation would also undermine the intended effect of redistribution and is therefore not desirable.

Note that the redistribution scheme must additionally handle details such as ensuring ALTO Clients retrieve ALTO information from the correct ALTO Server. See [I-D.gu-alto-redistribution] for further discussion. Details of a particular redistribution scheme are outside the scope of this document.

To fulfill these requirements, ALTO Information meant to be redistributable contains a digital signature which includes a hash of the ALTO information signed by the ALTO Server with its private key. The corresponding public key is included in the Server Information resource Section 7.7.1.1, along with the certificate chain to a Root Certificate generated by the ALTO Service Provider. To prevent man-in-the-middle attacks, an ALTO Client SHOULD perform the Server Information resource request over SSL/TLS and verify the server identity according to [RFC6125].

The signature verification algorithm is detailed in Section 8.1.3.3.

12.5. Denial of Service

ISPs should be cognizant of the workload at the ALTO Server generated by certain ALTO Queries, such as certain queries to the Map Filtering Service and Ranking Service. In particular, queries which can be generated with low effort but result in expensive workloads at the ALTO Server could be exploited for Denial-of-Service attacks. For instance, a simple ALTO query with n Source Network Locations and m Destination Network Locations can be generated fairly easily but results in the computation of n*m Path Costs between pairs by the ALTO Server (see Section 5.2). One way to limit Denial-of-Service attacks is to employ access control to the ALTO server. Another possible mechanism for an ALTO Server to protect itself against a multitude of computationally expensive bogus requests is to demand

that each ALTO Client to solve a computational puzzle first before allocating resources for answering a request (see, e.g., [I-D.jennings-sip-hashcash]). The current specification does not use such computational puzzles, and discussion regarding tradeoffs of such an approach would be needed before including such a technique in the ALTO Protocol.

ISPs should also leverage the fact that the the Map Service allows ALTO Servers to pre-generate maps that can be useful to many ALTO Clients.

12.6. ALTO Server Access Control

In order to limit access to an ALTO server (e.g., for an ISP to only allow its users to access its ALTO server, or to prevent Denial-of-Service attacks by arbitrary hosts from the Internet), an ALTO server may employ access control policies. Depending on the use-case and scenario, an ALTO server may restrict access to its services more strictly or rather openly (see [I-D.stiemerling-alto-deployments] for a more detailed discussion on this issue).

13. References

13.1. Normative References

[IEEE.754.2008] Institute of Electrical and Electronics Engineers, "Standard for Binary Floating-Point Arithmetic", IEEE Standard 754, August 2008.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

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

[RFC2616] 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.

[RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.

[RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, August 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, August 2010.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

13.2. Informative References

[BitTorrent]
"Bittorrent Protocol Specification v1.0", <http://wiki.theory.org/BitTorrentSpecification>.

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

[I-D.gu-alto-redistribution]
Yingjie, G., Alimi, R., and R. Even, "ALTO Information Redistribution", draft-gu-alto-redistribution-03 (work in progress), July 2010.

[I-D.ietf-alto-reqs]
Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO)

Requirements", draft-ietf-alto-reqs-08 (work in progress), March 2011.

[I-D.ietf-alto-server-discovery]
Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M., and S. Yongchao, "ALTO Server Discovery", draft-ietf-alto-server-discovery-00 (work in progress), May 2011.

[I-D.jennings-sip-hashcash]
Jennings, C., "Computational Puzzles for SPAM Reduction in SIP", draft-jennings-sip-hashcash-06 (work in progress), July 2007.

[I-D.mrw-nat66]
Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", draft-mrw-nat66-16 (work in progress), April 2011.

[I-D.p4p-framework]
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.

[I-D.saumitra-alto-multi-ps]
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.

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

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

[I-D.stiemerling-alto-deployments]
Stiemerling, M. and S. Kiesel, "ALTO Deployment Considerations", draft-stiemerling-alto-deployments-06 (work in progress), January 2011.

[I-D.wang-alto-p4p-specification]
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.

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

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.

Appendix A. Acknowledgments

Thank you to Jan Seedorf for contributions to the Security Considerations section. We would like to thank Yingjie Gu and Roni Even for helpful input and design concerning ALTO Information redistribution.

We would like to thank the following people whose input and involvement was indispensable in achieving this merged proposal:

Obi Akonjang (DT Labs/TU Berlin),

Saumitra M. Das (Qualcomm Inc.),

Syon Ding (China Telecom),

Doug Pasko (Verizon),

Laird Popkin (Pando Networks),

Satish Raghunath (Juniper Networks),

Albert Tian (Ericsson/Redback),

Yu-Shun Wang (Microsoft),

David Zhang (PPLive),

Yunfei Zhang (China Mobile).

We would also 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).

Appendix B. Authors

[[CmtAuthors: RFC Editor: Please move information in this section to the Authors' Addresses section at publication time.]]

Stefano Previdi
Cisco

Email: sprevidi@cisco.com

Stanislav Shalunov
BitTorrent

Email: shalunov@bittorrent.com

Richard Woundy
Comcast

Richard_Woundy@cable.comcast.com

Authors' Addresses

Richard Alimi (editor)
Google
1600 Amphitheatre Parkway
Mountain View CA
USA

Email: ralimi@google.com

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

Email: rpenno@juniper.net

Y. Richard Yang (editor)
Yale University
51 Prospect St
New Haven CT
USA

Email: yry@cs.yale.edu

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2012

R. Alimi, Ed.
Google
R. Penno, Ed.
Juniper Networks
Y. Yang, Ed.
Yale University
October 31, 2011

ALTO Protocol
draft-ietf-alto-protocol-10.txt

Abstract

Networking applications today already have access to a great amount of Inter-Provider network topology information. For example, views of the Internet routing table are easily available at looking glass servers and entirely practical to be downloaded by clients. What is missing is knowledge of the underlying network topology from the ISP or Content Provider (henceforth referred as Provider) point of view. In other words, what a Provider prefers in terms of traffic optimization -- and a way to distribute it.

The ALTO Service provides network information (e.g., basic network location structure, preferences of network paths) with the goal of modifying network resource consumption patterns while maintaining or improving application performance. The basic information of ALTO is based on abstract maps of a network. These maps provide simplified, yet enough information of a network for applications to effectively utilize. Additional services are built on top the maps.

This document describes a protocol implementing the ALTO Service. Although the ALTO 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 [RFC2119].

Status of this Memo

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

This Internet-Draft will expire on May 3, 2012.

Copyright Notice

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 (e.g., preferences) to applications. On the other hand, modern network applications can be adaptive, with the potential to become more network-efficient (e.g., reduce network resource consumption) and achieve better application performance (e.g., accelerated download rate), by leveraging better network-provided information.

The ALTO Service intends to provide a simple mechanism to convey network information to applications. Its objective is to provide basic, abstract but useful network information to applications. The mechanism should include abstractions to achieve concise, flexible network information expression.

The goal of this document is to specify a simple and unified protocol that meets the ALTO requirements [I-D.ietf-alto-reqs] while 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

- P4P [I-D.p4p-framework], [P4P-SIGCOMM08], [I-D.wang-alto-p4p-specification];
- ALTO Info-Export [I-D.shalunov-alto-infoexport];
- Query/Response [I-D.saumitra-alto-queryresponse], [I-D.saumitra-alto-multi-ps];
- ATTP [ATTP];
- Proxidor [I-D.akonjang-alto-proxidor].

See Appendix A for a list of people that have contributed significantly to this effort and the projects and proposals listed above.

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 peer selection process in distributed applications in order to increase locality of traffic, improve user-experience, amongst others. It also helps ISPs to efficiently manage traffic that traverses more expensive links such as transit and backup links, thus allowing a better provisioning of the networking infrastructure.

1.3.2. 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.

An example type of application is a Peer-to-Peer overlay where peer selection can be improved by including ALTO information in the selection process.

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. This section first introduces the terminology, and then defines the ALTO architecture and the ALTO Protocol's place in the overall architecture.

2.1. Terminology

We use the following terms defined in [RFC5693]: 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, Autonomous System Number (ASN), and Network Location.

2.1.1. Endpoint Address

An endpoint address represents the communication address of an endpoint. 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.

Each Endpoint Address has an associated Address Type, which indicates both its syntax and semantics.

2.1.2. ASN

An Autonomous System Number.

2.1.3. Network Location

Network Location is a generic term denoting a single endpoint or group of endpoints.

2.1.4. ALTO Information

ALTO Information is a generic term referring to the network information sent by an ALTO Server.

2.1.5. ALTO Information Base

Internal representation of the ALTO Information maintained by the ALTO Server. Note that the structure of this internal representation is not defined by this document.

2.2. ALTO Service and Protocol Scope

An ALTO Server conveys the network information from the perspective of a network region; the ALTO Server presents its "my-Internet View" of the network region. In particular, an ALTO Server defines network Endpoints (and aggregations thereof) and generic costs amongst them, both from the network region's own perspective. A network region in this context can be an Autonomous System, an ISP, or perhaps a smaller region or 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 Server prepares ALTO Information; an ALTO Client uses ALTO Service Discovery to identify an appropriate ALTO Server; 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. The ALTO Server aggregates information from multiple systems to provide an abstract, unified, useful network view to applications. Examples of other systems 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.

Note that it may also be possible for ALTO Servers to exchange network information with other ALTO Servers (either within the same administrative domain or another administrative domain with the consent of both parties) in order to adjust exported ALTO Information. Such a protocol is also outside the scope of this specification.

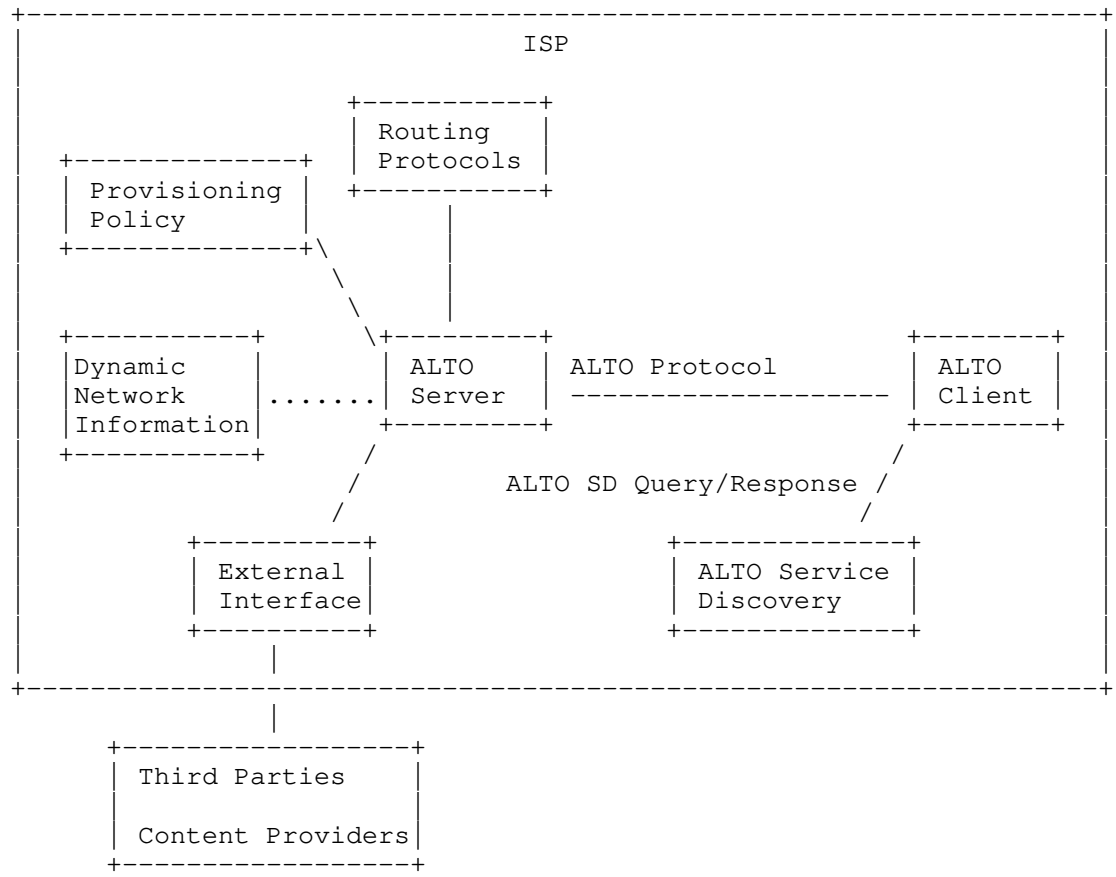


Figure 1: Basic ALTO Architecture.

3. Protocol Structure

The ALTO Protocol uses a simple extensible framework to convey network information. In the general framework, the ALTO protocol will convey properties on both Network Locations and the paths between Network Locations.

In this document, we focus on a particular Endpoint property to denote the location of an endpoint, and provider-defined costs for paths between pairs of Network Locations.

The ALTO Protocol is built on a common transport protocol, messaging structure and encoding, and transaction model. The protocol is subdivided into services of related functionality. ALTO-Core

provides the Server Information Service and the Map Service to provide ALTO Information. Other ALTO Information services can provide additional functionality. There are three such services defined in this document: the Map Filtering Service, Endpoint Property Service, and Endpoint Cost Service. Additional services may be defined in companion documents. Note that functionality offered in different services are not totally non-overlapping (e.g., the Map Service and Map Filtering Service).

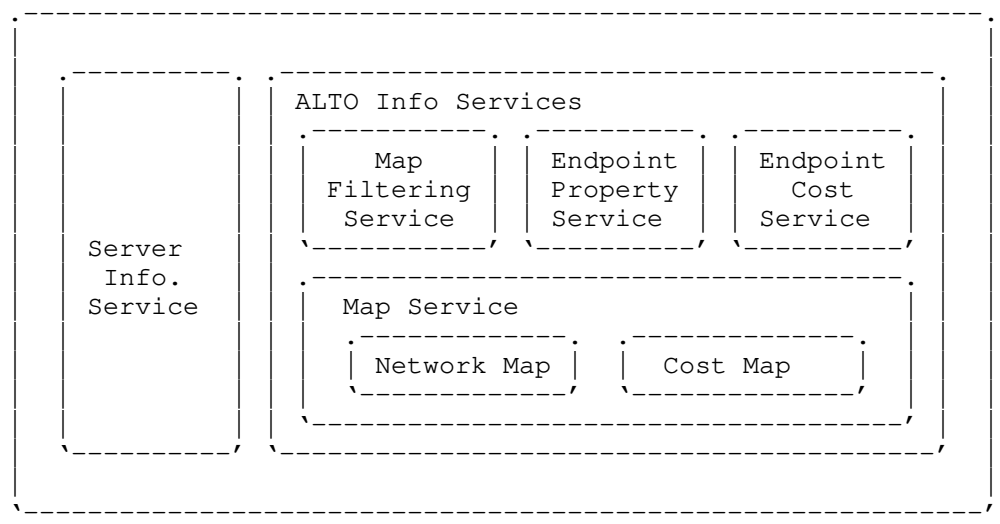


Figure 2: ALTO Protocol Structure

3.1. Server Information Service

The Server Information Service lists the details on the information that can be provided by an ALTO Server and perhaps other ALTO Servers maintained by the network provider. The configuration includes, for example, details about the operations and cost metrics supported by the ALTO Server and other related ALTO Servers that may be usable by an ALTO Client.

3.2. ALTO Information Services

Multiple, distinct services are defined to allow ALTO Clients to query ALTO Information from an ALTO Server. The ALTO Server internally maintains an ALTO Information Base that encodes the network provider's preferences. The ALTO Information Base encodes the Network Locations defined by the ALTO Server (and their corresponding properties), as well as the provider-defined costs

between pairs of Network Locations.

3.2.1. Map Service

The Map Service provides batch information to ALTO Clients in the form of Network Map and Cost Map. The Network Map (See Section 4) provides the full set of Network Location groupings defined by the ALTO Server and the endpoints contained with each grouping. The Cost Map (see Section 5) provides costs between the defined groupings.

These two maps can be thought of (and implemented as) as simple files with appropriate encoding provided by the ALTO Server.

3.2.2. Map Filtering Service

Resource constrained ALTO Clients may benefit from query results being filtered at the ALTO Server. This avoids an ALTO Client spending network bandwidth or CPU collecting results and performing client-side filtering. The Map Filtering Service allows ALTO Clients to query for the ALTO Server Network Map and Cost Map based on additional parameters.

3.2.3. Endpoint Property Service

This service allows ALTO Clients to look up properties for individual endpoints. An example endpoint property is its Network Location (its grouping defined by the ALTO Server) or connectivity type (e.g., ADSL, Cable, or FTTH).

3.2.4. Endpoint Cost Service

Some ALTO Clients may also benefit from querying for costs and rankings based on endpoints. The Endpoint Cost Service allows an ALTO Server to return either numerical costs or ordinal costs (rankings) directly amongst Endpoints.

4. Network Map

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 losing 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 definition of proximity varies depending on the granularity of the ALTO information configured by the provider. 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.

As used in this document, the Network Map refers to the syntax and semantics of the information distributed by the ALTO Server. This document does not discuss the internal representation of this data structure within the ALTO Server.

4.1. PID

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

A PID is an identifier that provides an indirect and network-agnostic way to specify an aggregation of network endpoints that may be treated similarly, based on network topology, type, or other properties. 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 and can improve scalability. In particular, network preferences (costs) may be specified between PIDs, allowing cost information to be more compactly represented and updated at a faster time scale than the network aggregations themselves.

Using PIDs, the Network Map may also be used to communicate simple preferences with only minimal information from the Cost Map. 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. The ISP may aggregate endhosts within a PoP into a single PID in the Network Map. The Cost Map may be encoded to indicate that peering within the same PID is preferred; for example, cost(PID_i, PID_i) == c* and cost(PID_i, PID_j) > c* for i != j. Section 5 provides further details about Cost Map structure.

4.2. Endpoint Addresses

Communicating endpoints may have many types of addresses, such as IP addresses, MAC addresses, or overlay IDs. The current specification only considers IP addresses.

4.2.1. IP Addresses

The endpoints aggregated into a PID are denoted by a list of IP prefixes. When either an ALTO Client or ALTO Server needs to determine which PID in a Network Map contains a particular IP address, longest-prefix matching MUST be used.

A Network Map MUST define a PID for each possible address in the IP address space for all of the address types contained in the map. A RECOMMENDED way to satisfy this property is to define a PID that contains the 0.0.0.0/0 prefix for IPv4 or ::/0 (for IPv6).

Each endpoint MUST map into exactly one PID. Since longest-prefix matching is used to map an endpoint to a PID, this can be accomplished by ensuring that no two PIDs contain an identical IP prefix.

4.3. Example Network Map

Figure 3 illustrates an example Network Map. PIDs are used to identify network-agnostic aggregations.

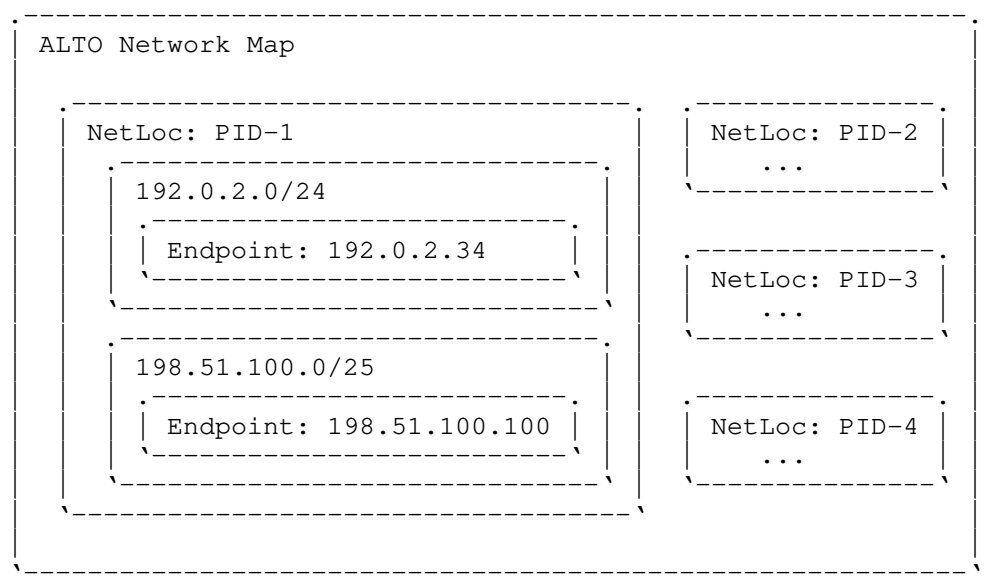


Figure 3: Example Network Map

5. Cost Map

An ALTO Server indicates preferences amongst network locations in the form of Path Costs. Path Costs are generic costs and can be internally computed by a network provider according to its own needs.

An ALTO Cost Map defines Path Costs pairwise amongst sets of source and destination Network Locations.

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.

As used in this document, the Cost Map refers to the syntax and semantics of the information distributed by the ALTO Server. This document does not discuss the internal representation of this data structure within the ALTO Server.

5.1. Cost Attributes

Path Costs have attributes:

- Type: identifies what the costs represent;
- Mode: identifies how the costs should be interpreted.

Certain queries for Cost Maps allow the ALTO Client to indicate the desired Type and Mode.

5.1.1. 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 strings.

5.1.1.1. Cost Type: routingcost

An ALTO Server MUST define the 'routingcost' Cost Type.

This Cost Type conveys a generic measure for the cost of routing traffic from a source to a destination. Lower values indicate a higher preference for traffic to be sent from a source to a destination.

Note that an ISP may internally compute routing cost using any method

it chooses (e.g., air-miles or hop-count) as long as it conforms to these semantics.

5.1.2. Cost Mode

The Mode attribute indicates how costs should be interpreted. Specifically, the Mode attribute indicates whether returned costs 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 that would make sense for numerical values, do not make sense for ordinal rankings. ALTO Clients may handle such costs differently.

Cost Modes are indicated in protocol messages as strings.

An ALTO Server MUST support at least one of 'numerical' and 'ordinal' costs. ALTO Clients SHOULD be cognizant of operations when a desired cost mode is not supported. For example, an ALTO Client desiring numerical costs may adjust behavior if only the ordinal Cost Mode is available. Alternatively, an ALTO Client desiring ordinal costs may construct ordinal costs given numerical values if only the numerical Cost Mode is available.

5.1.2.1. Cost Mode: numerical

This Cost Mode is indicated by the string 'numerical'. This mode indicates that it is safe to perform numerical operations (e.g. normalization) on the returned costs.

5.1.2.2. Cost Mode: ordinal

This Cost Mode is indicated by the string 'ordinal'. This mode indicates that the costs values to a set of Destination Network Locations from a particular Source Network Location are a ranking, with lower values indicating a higher preference. The values are non-negative integers. Ordinal cost values from a particular Source Network Location to a set of Destination Network Locations need not be unique nor contiguous. In particular, from the perspective of a particular Source Network Location, two Destination Network Locations may have an identical rank (ordinal cost value). This document does not specify any behavior by an ALTO Client in this case; an ALTO Client may decide to break ties by random selection, other application knowledge, or some other means.

It is important to note that the values in the Cost Map provided with the ordinal Cost Mode are not necessarily the actual cost known to the ALTO Server.

5.2. Cost Map Structure

A query for a Cost Map either explicitly or implicitly includes a list of Source Network Locations and a list of Destination Network Locations. (Recall that a Network Location can be an endpoint address or a PID.)

Specifically, assume that a 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].

The ALTO Server will return 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). If the ALTO Server does not define a Path Cost for a particular pair, it may be omitted. We refer to this structure as a Cost Map.

If the Cost Mode is 'ordinal', the Path Cost of each communicating pair is relative to the m*n entries.

5.3. Network Map and Cost Map Dependency

If a Cost Map contains PIDs in the list of Source Network Locations or the list of Destination Network Locations, the Path Costs are generated based on a particular Network Map (which defines the PIDs). Version Tags are introduced to ensure that ALTO Clients are able to use consistent information even though the information is provided in two maps.

A Version Tag is an opaque string associated with a Network Map maintained by the ALTO Server. When the Network Map changes, the Version Tag MUST also be changed. (Thus, the Version Tag is defined similarly to HTTP's Entity Tags; see Section 3.11 of [RFC2616].) Possibilities for generating a Version Tag include the last-modified timestamp for the Network Map, or a hash of its contents.

A Network Map distributed by the ALTO Server includes its Version Tag. A Cost Map referring to PIDs also includes the Version Tag of the Network Map on which it is based.

6. Protocol Design Overview

The ALTO Protocol design uses a REST-ful design with the goal of leveraging current HTTP [RFC2616] implementations and infrastructure. The REST-ful design supports flexible deployment strategies and provides extensibility. ALTO requests and responses are encoded with JSON [RFC4627].

6.1. Benefits

Benefits enabled by these design choices include easier understanding and debugging, mature libraries, tools, infrastructure, and caching and redistribution of ALTO information for increased scalability.

6.1.1. Existing Infrastructure

HTTP is a natural choice for integration with existing applications and infrastructure. In particular, the ALTO Protocol design leverages:

- the huge installed base of infrastructure, including HTTP caches,
- mature software implementations,
- the fact that many P2P clients already have an embedded HTTP client, and
- authentication and encryption mechanisms in HTTP and SSL/TLS.

6.1.2. ALTO Information Reuse and Redistribution

ALTO information may be useful to a large number of applications and users. For example, an identical Network Map may be used by all ALTO Clients querying a particular ALTO Server. At the same time, distributing ALTO information must be efficient and not become a bottleneck.

Beyond integration with existing HTTP caching infrastructure, ALTO information may also be cached or redistributed using application-dependent mechanisms, such as P2P DHTs or P2P file-sharing. This document does not define particular mechanisms for such redistribution, but it does define the primitives (e.g., digital signatures) needed to support such a mechanism. See [I-D.gu-alto-redistribution] for further discussion.

Note that if caching or redistribution is used, the response message may be returned from another (possibly third-party) entity. Reuse and Redistribution is further discussed in Section 12.4. Protocol

support for redistribution is specified in Section 8.

6.2. Protocol Design

The ALTO Protocol uses a REST-ful design. There are two primary components to this design:

- Information Resources: Each service provides network information as a set of resources, which are distinguished by their media types [RFC2046]. An ALTO Client may construct an HTTP request for a particular resource (including any parameters, if necessary), and an ALTO Server returns the requested resource in an HTTP response.

- Information Resource Directory: An ALTO Server provides to ALTO Clients a list of available resources and the URI at which each is provided. This document refers to this list as the Information Resource Directory. This directory is the single entry point to an ALTO Service. ALTO Clients consult the directory to determine the services provided by an ALTO Server.

7. Protocol Specification

This section first specifies general client and server processing, followed by a detailed specification for each ALTO Information Resource.

7.1. Notation

This document uses an adaptation of the C-style struct notation to define the required and optional members of JSON objects. Unless explicitly noted, each member of a struct is REQUIRED.

The types 'JSONString', 'JSONNumber', 'JSONBool' indicate the JSON string, number, and boolean types, respectively.

Note that no standard, machine-readable interface definition or schema is provided. Extension documents may document these as necessary.

7.2. Basic Operation

The ALTO Protocol employs standard HTTP [RFC2616]. It is used for discovering available Information Resources at an ALTO Server and retrieving Information Resources. ALTO Clients and ALTO Servers use HTTP requests and responses carrying ALTO-specific content with encoding as specified in this document, and MUST be compliant with

[RFC2616].

7.2.1. Discovering Information Resources

To discover available resources, an ALTO Client may request the Information Resource Directory, which an ALTO Server provides at the URI found by the ALTO Discovery protocol.

Informally, an Information Resource Directory enumerates URIs at which an ALTO Server offers Information Resources. Each entry in the directory indicates a URI at which an ALTO Server accepts requests, and returns either the requested Information Resource or an Information Resource Directory that references additional Information Resources. See Section 7.6 for a detailed specification.

7.2.2. Requesting Information Resources

Through the retrieved Information Resource Directories, an ALTO Client can determine whether an ALTO Server supports the desired Information Resource, and if it is supported, the URI at which it is available.

Where possible, the ALTO Protocol uses the HTTP GET method to request resources. However, some ALTO services provide Information Resources that are the function of one or more input parameters. Input parameters are encoded in the HTTP request's entity body, and the request uses the HTTP POST method.

Note that it is possible for an ALTO Server to employ caching for the response to a POST request. This can be accomplished by returning an HTTP 303 status code ("See Other") indicating to the ALTO Client that the resulting Cost Map is available via a GET request to an alternate URL (which may be cached).

When requesting an ALTO Information Resource that requires input parameters specified in a HTTP POST request, an ALTO Client MUST set the Content-Type HTTP header to the media type corresponding to the format of the supplied input parameters.

7.2.3. Response

Upon receiving a request, an ALTO server either returns the requested resource, provides the ALTO Client an Information Resource Directory indicating how to reach the desired resource, or returns an error.

The type of response MUST be indicated by the media type attached to the response (the Content-Type HTTP header). If an ALTO Client receives an Information Resource Directory, it can consult the

received directory to determine if any of the offered URIs contain the desired Information Resource.

The generic encoding for an Information Resource is specified in Section 7.3.

Errors are indicated via either ALTO-level error codes, or via HTTP status codes; see Section 7.4.

7.2.4. Client Behavior

7.2.4.1. Using Information Resources

This specification does not indicate any required actions taken by ALTO Clients upon successfully receiving an Information Resource from an ALTO Server. Although ALTO Clients are suggested to interpret the received ALTO Information and adapt application behavior, ALTO Clients are not required to do so.

7.2.4.2. Error Conditions

If an ALTO Client does not successfully receive a desired Information Resource from a particular ALTO Server, it can either choose another server (if one is available) or fall back to a default behavior (e.g., perform peer selection without the use of ALTO information). An ALTO Client may also retry the request at a later time.

7.2.5. Authentication and Encryption

An ALTO Server MAY support SSL/TLS to implement server and/or client authentication, as well as encryption. See [RFC6125] for considerations regarding verification of server identity.

7.2.6. HTTP Cookies

If cookies are included in an HTTP request received by an ALTO Server, they MUST be ignored.

7.2.7. Parsing

This document only details object members used by this specification. Extensions may include additional members within JSON objects defined in this document. ALTO implementations MUST ignore such unknown fields when processing ALTO messages.

7.3. Information Resource

An Information Resource is an HTTP entity body received by an ALTO Server that encodes the ALTO Information desired by an ALTO Client.

This document specifies multiple Information Resources that can be provided by an ALTO Server. Each Information Resource has certain attributes associated with it, indicating its data format, the input parameters it supports, and format of the input parameters.

7.3.1. Capabilities

An ALTO Server may advertise to an ALTO Client that it supports certain capabilities in requests for an Information Resource. For example, if an ALTO Server allows requests for a Cost Map to include constraints, it may advertise that it supports this capability.

7.3.2. Input Parameters Media Type

An ALTO Server may allow an ALTO Client to supply input parameters when requesting certain Information Resources. The format of the input parameters (i.e., as contained in the entity body of the HTTP POST request) is indicated by the media type [RFC2046].

7.3.3. Media Type

The media type [RFC2046] uniquely indicates the data format of the Information Resource as returned by an ALTO Server in the HTTP entity body.

7.3.4. Encoding

Though each Information Resource may have a distinct syntax, they are designed to have a common structure containing generic ALTO-layer metadata about the resource, as well as data itself.

An Information Resource has a single top-level JSON object of type InfoResourceEntity:

```
object {
  InfoResourceMetaData   meta;
  [InfoResourceDataType] data;
} InfoResourceEntity;
```

with members:

meta meta-information pertaining to the Information Resource

data the data contained in the Information Resource

7.3.4.1. Meta Information

Meta information is encoded as a JSON object with type InfoResourceMetaData:

```
object {
  InfoResourceRedistDesc redistribution;    [OPTIONAL]
} InfoResourceMetaData;
```

with members:

redistribution Additional data for use in Information Resources that may be redistributed amongst ALTO Clients. See Section 8.

7.3.4.2. ALTO Information

The "data" member of the InfoResourceEntity encodes the resource-specific data; the structure of this member is detailed later in this section for each particular Information Resource.

7.3.4.3. Signature

An ALTO Server MAY additionally supply a signature asserting that it generated a particular response. See Section 8.2.2.

7.3.4.4. Example

The following is an example of the encoding for an Information Resource:

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {
    "redistribution" : { ... }
  },
  "data" : {
    ...
  }
```

```
}
```

7.4. ALTO Errors

If there is an error processing a request, an ALTO Server SHOULD return additional ALTO-layer information, if it is available, in the form of an ALTO Error Resource encoded in the HTTP response's entity body.

If no ALTO-layer information is available, an ALTO Server may omit an ALTO Error resource from the response. An appropriate HTTP status code MUST be set.

It is important to note that the HTTP Status Code and ALTO Error Code have distinct roles. An ALTO Error Code provides detailed information about the why a particular request for an ALTO Resource was not successful. The HTTP status code indicates to HTTP processing elements (e.g., intermediaries and clients) how the response should be treated.

7.4.1. Media Type

The media type for an Error Resource is "application/alto-error+json".

7.4.2. Resource Format

An Error Resource has the format:

```
object {
  JSONString code;
  JSONString reason;    [OPTIONAL]
} ErrorResourceEntity;
```

where:

code An ALTO Error Code defined in Table 1

reason A (free-form) human-readable explanation of the particular error

7.4.3. Error Codes

This document defines ALTO Error Codes to support the error conditions needed for purposes of this document. Additional status codes may be defined in companion or extension documents.

The HTTP status codes corresponding to each ALTO Error Code are defined to provide correct behavior with HTTP intermediaries and clients. When an ALTO Server returns a particular ALTO Error Code, it MUST indicate one of the corresponding HTTP status codes in Table 1in the HTTP response.

If multiple errors are present in a single request (e.g., a request uses a JSONString when a JSONInteger is expected and a required field is missing), then the ALTO Server MUST return exactly one of the detected errors. However, the reported error is implementation defined, since specifying a particular order for message processing encroaches needlessly on implementation technique.

ALTO Error Code	HTTP Status Code(s)	Description
E_SYNTAX	400	Parsing error in request (including identifiers)
E_JSON_FIELD_MISSING	400	Required field missing
E_JSON_VALUE_TYPE	400	JSON Value of unexpected type
E_INVALID_COST_MODE	400	Invalid cost mode
E_INVALID_COST_TYPE	400	Invalid cost type
E_INVALID_PROPERTY_TYPE	400	Invalid property type

Table 1: Defined ALTO Error Codes

7.5. ALTO Types

This section details the format for particular data values used in the ALTO Protocol.

7.5.1. PID Name

A PID Name is encoded as a US-ASCII string. The string MUST be no more than 64 characters, and MUST NOT contain any ASCII character below 0x21 or above 0x7E or the '.' separator. The '.' separator is reserved for future use and MUST NOT be used unless specifically indicated by a companion or extension document.

The type 'PIDName' is used in this document to indicate a string of this format.

7.5.2. Version Tag

A Version Tag is encoded as a US-ASCII string. The string MUST be no more than 64 characters, and MUST NOT contain any ASCII character below 0x21 or above 0x7E.

The type 'VersionTag' is used in this document to indicate a string of this type.

7.5.3. Endpoints

This section defines formats used to encode addresses for Endpoints. In a case that multiple textual representations encode the same Endpoint address or prefix (within the guidelines outlined in this document), the ALTO Protocol does not require ALTO Clients or ALTO Servers to use a particular textual representation, nor does it require that ALTO Servers reply to requests using the same textual representation used by requesting ALTO Clients. ALTO Clients must be cognizant of this.

7.5.3.1. Address Type

Address Types are encoded as US-ASCII strings consisting of only alphanumeric characters. This document defines the address type "ipv4" to refer to IPv4 addresses, and "ipv6" to refer to IPv6 addresses. Extension documents may define additional Address Types.

The type 'AddressType' is used in this document to indicate a string of this format.

7.5.3.2. Endpoint Address

Endpoint Addresses are encoded as US-ASCII strings. The exact characters and format depend on the type of endpoint address.

The type 'EndpointAddr' is used in this document to indicate a string of this format.

7.5.3.2.1. IPv4

IPv4 Endpoint Addresses are encoded as specified by the 'IPv4address' rule in Section 3.2.2 of [RFC3986].

7.5.3.2.2. IPv6

IPv6 Endpoint Addresses are encoded as specified in Section 4 of [RFC5952].

7.5.3.2.3. Typed Endpoint Addresses

When an Endpoint Address is used, an ALTO implementation must be able to determine its type. For this purpose, the ALTO Protocol allows endpoint addresses to also explicitly indicate their type.

Typed Endpoint Addresses are encoded as US-ASCII strings of the format 'AddressType:EndpointAddr' (with the ':' character as a separator). The type 'TypedEndpointAddr' is used to indicate a string of this format.

7.5.3.3. Endpoint Prefixes

For efficiency, it is useful to denote a set of Endpoint Addresses using a special notation (if one exists). This specification makes use of the prefix notations for both IPv4 and IPv6 for this purpose.

Endpoint Prefixes are encoded as US-ASCII strings. The exact characters and format depend on the type of endpoint address.

The type 'EndpointPrefix' is used in this document to indicate a string of this format.

7.5.3.3.1. IPv4

IPv4 Endpoint Prefixes are encoded as specified in Section 3.1 of [RFC4632].

7.5.3.3.2. IPv6

IPv6 Endpoint Prefixes are encoded as specified in Section 7 of [RFC5952].

7.5.3.4. Endpoint Address Group

The ALTO Protocol includes messages that specify potentially large sets of endpoint addresses. Endpoint Address Groups provide a more efficient way to encode such sets, even when the set contains

endpoint addresses of different types.

An Endpoint Address Group is defined as:

```
object {
  EndpointPrefix [AddressType]<0..*>;
  ...
} EndpointAddrGroup;
```

In particular, an Endpoint Address Group is a JSON object with the name of each member being the string corresponding to the address type, and the member's corresponding value being a list of prefixes of addresses of that type.

The following is an example with both IPv4 and IPv6 endpoint addresses:

```
{
  "ipv4": [
    "192.0.2.0/24",
    "198.51.100.0/25"
  ],
  "ipv6": [
    "2001:db8:0:1::/64",
    "2001:db8:0:2::/64"
  ]
}
```

7.5.4. Cost Mode

A Cost Mode is encoded as a US-ASCII string. The string MUST either have the value 'numerical' or 'ordinal'.

The type 'CostMode' is used in this document to indicate a string of this format.

7.5.5. Cost Type

A Cost Type is encoded as a US-ASCII string. The string MUST be no more than 32 characters, and MUST NOT contain characters other than alphanumeric characters, the hyphen ('-'), or the ':' separator.

Identifiers prefixed with 'priv:' are reserved for Private Use [RFC5226]. Identifiers prefixed with 'exp:' are reserved for

Experimental use. All other identifiers appearing in an HTTP request or response with an 'application/alto-*' media type MUST be registered in the ALTO Cost Types registry Section 11.2.

The type 'CostType' is used in this document to indicate a string of this format.

7.5.6. Endpoint Property

An Endpoint Property is encoded as a US-ASCII string. The string MUST be no more than 32 characters, and MUST NOT contain characters other than alphanumeric characters, the hyphen ('-'), or the ':' separator.

Identifiers prefixed with 'priv:' are reserved for Private Use [RFC5226]. Identifiers prefixed with 'exp:' are reserved for Experimental use. All other identifiers appearing in an HTTP request or response with an 'application/alto-*' media type MUST be registered in the ALTO Endpoint Property registry Section 11.3.

The type 'EndpointProperty' is used in this document to indicate a string of this format.

7.6. Information Resource Directory

An Information Resource Directory indicates to ALTO Clients which Information Resources are made available by an ALTO Server.

Since resource selection happens after consumption of the Information Resource Directory, the format of the Information Resource Directory is designed to be simple with the intention of future ALTO Protocol versions maintaining backwards compatibility. Future extensions or versions of the ALTO Protocol SHOULD be accomplished by extending existing media types or adding new media types, but retaining the same format for the Information Resource Directory.

An ALTO Server MUST make an Information Resource Directory available via the HTTP GET method to a URI discoverable by an ALTO Client. Discovery of this URI is out of scope of this document, but could be accomplished by manual configuration or by returning the URI of an Information Resource Directory from the ALTO Discovery Protocol [I-D.ietf-alto-server-discovery].

7.6.1. Media Type

The media type is "application/alto-directory+json".

7.6.2. Encoding

An Information Resource Directory is a JSON object of type InfoResourceDirectory:

```
object {
  ...
} Capabilities;

object {
  JSONString   uri;
  JSONString   media-types<1..*>;
  JSONString   accepts<0..*>;        [OPTIONAL]
  Capabilities capabilities;         [OPTIONAL]
} ResourceEntry;

object {
  ResourceEntry resources<0..*>;
} InfoResourceDirectory;
```

where the "resources" array indicates a list of Information Resources provided by an ALTO Server. Note that the list of available resources is enclosed in a JSON object for extensibility; future protocol versions may specify additional members in the InfoResourceDirectory object.

Any URI endpoint indicated in an Information Resource Directory MAY provide a response to an OPTIONS request that is in the format of an Information Resource Directory. This provides ALTO Clients a means to discover resources and capabilities offered by that URI endpoint. ALTO Servers that reply with an HTTP 300 status code ("Multiple Choices") SHOULD use the Information Resource Directory format in the reply.

Each entry in the directory specifies:

uri A URI at which the ALTO Server provides one or more Information Resources, or an Information Resource Directory indicating additional Information Resources.

media-types The list of all media types of Information Resources (see Section 7.3.3) available via GET or POST requests to the corresponding URI or URIs discoverable via the URI.

accepts The list of all media types of input parameters (see Section 7.3.2) accepted by POST requests to the corresponding URI or URIs discoverable via the URI. If this member is not present, it MUST be assumed to be an empty array.

capabilities A JSON Object enumerating capabilities of an ALTO Server in providing the Information Resource at the corresponding URI and Information Resources discoverable via the URI. If this member is not present, it MUST be assumed to be an empty array. If a capability for one of the offered Information Resources is not explicitly listed here, an ALTO Client may either issue an OPTIONS HTTP request to the corresponding URI to determine if the capability is supported, or assume its default value.

If an entry has an empty list for "accepts", then the corresponding URI MUST support GET requests. If an entry has a non-empty list for "accepts", then the corresponding URI MUST support POST requests. If an ALTO Server wishes to support both GET and POST on a single URI, it MUST specify two entries in the Information Resource Directory.

7.6.3. Example

The following is an example Information Resource Directory returned by an ALTO Server. In this example, the ALTO Server provides additional Network and Cost Maps via a separate subdomain, "custom.alto.example.com". The maps available via this subdomain are Filtered Network and Cost Maps as well as pre-generated maps for the "hopcount" and "routingcost" Cost Types in the "ordinal" Cost Mode.

An ALTO Client can discover the maps available by "custom.alto.example.com" by successfully performing an OPTIONS request to "http://custom.alto.example.com/maps".

```
GET /directory HTTP/1.1
Host: alto.example.com
Accept: application/alto-directory+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-directory+json

{
  "resources" : [
    {
```

```
        "uri" : "http://alto.example.com/serverinfo",
        "media-types" : [ "application/alto-serverinfo+json" ]
      }, {
        "uri" : "http://alto.example.com/networkmap",
        "media-types" : [ "application/alto-networkmap+json" ]
      }, {
        "uri" : "http://alto.example.com/costmap/num/routingcost",
        "media-types" : [ "application/alto-costmap+json" ],
        "capabilities" : {
          "cost-modes" : [ "numerical" ],
          "cost-types" : [ "routingcost" ]
        }
      }, {
        "uri" : "http://alto.example.com/costmap/num/hopcount",
        "media-types" : [ "application/alto-costmap+json" ],
        "capabilities" : {
          "cost-modes" : [ "numerical" ],
          "cost-types" : [ "hopcount" ]
        }
      }, {
        "uri" : "http://custom.alto.example.com/maps",
        "media-types" : [
          "application/alto-networkmap+json",
          "application/alto-costmap+json"
        ],
        "accepts" : [
          "application/alto-networkmapfilter+json",
          "application/alto-costmapfilter+json"
        ]
      }, {
        "uri" : "http://alto.example.com/endpointprop/lookup",
        "media-types" : [ "application/alto-endpointprop+json" ],
        "accepts" : [ "application/alto-endpointpropparams+json" ],
        "capabilities" : {
          "prop-types" : [ "pid" ]
        }
      }, {
        "uri" : "http://alto.example.com/endpointcost/lookup",
        "media-types" : [ "application/alto-endpointcost+json" ],
        "accepts" : [ "application/alto-endpointcostparams+json" ],
        "capabilities" : {
          "cost-constraints" : true,
          "cost-modes" : [ "ordinal", "numerical" ],
          "cost-types" : [ "routingcost", "hopcount" ]
        }
      }
    ]
  }
```

```
OPTIONS /maps HTTP/1.1
Host: custom.alto.example.com
Accept: application/alto-directory+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-directory+json

{
  "resources" : [
    {
      "uri" : "http://custom.alto.example.com/networkmap/filtered",
      "media-types" : [ "application/alto-networkmap+json" ],
      "accepts" : [ "application/alto-networkmapfilter+json" ]
    }, {
      "uri" : "http://custom.alto.example.com/costmap/filtered",
      "media-types" : [ "application/alto-costmap+json" ],
      "accepts" : [ "application/alto-costmapfilter+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "cost-modes" : [ "ordinal", "numerical" ],
        "cost-types" : [ "routingcost", "hopcount" ]
      }
    }, {
      "uri" : "http://custom.alto.example.com/ord/routingcost",
      "media-types" : [ "application/alto-costmap+json" ],
      "capabilities" : {
        "cost-modes" : [ "ordinal" ],
        "cost-types" : [ "routingcost" ]
      }
    }, {
      "uri" : "http://custom.alto.example.com/ord/hopcount",
      "media-types" : [ "application/alto-costmap+json" ],
      "capabilities" : {
        "cost-modes" : [ "ordinal" ],
        "cost-types" : [ "hopcount" ]
      }
    }
  ]
}
```

7.6.4. Usage Considerations

7.6.4.1. ALTO Client

This document specifies no requirements or constraints on ALTO Clients with regards to how they process an Information Resource Directory to identify the URI corresponding to a desired Information Resource. However, some advice is provided for implementors.

It is possible that multiple entries in the directory match a desired Information Resource. For instance, in the example in Section 7.6.3, a full Cost Map with "numerical" Cost Mode and "routingcost" Cost Type could be retrieved via a GET request to "http://alto.example.com/costmap/num/routingcost", or via a POST request to "http://custom.alto.example.com/costmap/filtered".

In general, it is preferred for ALTO Clients to use GET requests where appropriate, since it is more likely for responses to be cacheable.

7.6.4.2. ALTO Server

This document indicates that an ALTO Server may or may not provide the Information Resources specified in the Map Filtering Service. If these resources are not provided, it is indicated to an ALTO Client by the absence of a Network Map or Cost Map with any media types listed under "accepts".

7.7. Information Resources

This section documents the individual Information Resources defined in the ALTO Protocol.

7.7.1. Server Information Service

The Server Information Service provides generic information about an ALTO Server.

7.7.1.1. Server Info

This Information Resource MUST be provided by an ALTO Server.

7.7.1.1.1. Media Type

The media type is "application/alto-serverinfo+json".

7.7.1.1.2. HTTP Method

This resource is requested using the HTTP GET method.

7.7.1.1.3. Input Parameters

None.

7.7.1.1.4. Capabilities

None.

7.7.1.1.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceServerInfo:

```
object {
  JSONString   service-id;             [OPTIONAL]
  JSONString   certificates<0..*>;     [OPTIONAL]
} InfoResourceServerInfo;
```

which has members:

service-id UUID [RFC4122] indicating an one or more ALTO Servers serving equivalent ALTO Information.

certificates List of PEM-encoded X.509 certificates used by the ALTO Server in the signing of responses.

If an ALTO Server has the possibility of marking any response as redistributable, the 'service-id' and 'certificates' fields are REQUIRED instead of OPTIONAL. See Section 8&$160;for detailed specification.

7.7.1.1.6. Example

```
GET /serverinfo HTTP/1.1
Host: alto.example.com
Accept: application/alto-serverinfo+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-serverinfo+json

{
  "meta" : {},
  "data" : {
    "service-id" : "c89ca72f-dead-41b5-9e2b-b65455ace1ee",
    "certificates" : [ ... ]
  }
}
```

7.7.2. Map Service

The Map Service provides batch information to ALTO Clients in the form of two types of maps: a Network Map and Cost Map.

7.7.2.1. Network Map

The Network Map Information Resource lists for each PID, the network locations (endpoints) within the PID. It MUST be provided by an ALTO Server.

7.7.2.1.1. Media Type

The media type is "application/alto-networkmap+json".

7.7.2.1.2. HTTP Method

This resource is requested using the HTTP GET method.

7.7.2.1.3. Input Parameters

None.

7.7.2.1.4. Capabilities

None.

7.7.2.1.5. Response

The returned InfoResourceEntity object "data" member of type InfoResourceNetworkMap:

```
object {
  EndpointAddrGroup [pidname]<0..*>;
  ...
} NetworkMapData;

object {
  VersionTag     map-vtag;
  NetworkMapData map;
} InfoResourceNetworkMap;
```

with members:

map-vtag The Version Tag (Section 5.3) of the Network Map.

map The Network Map data itself.

NetworkMapData is a JSON object with each member representing a single PID and its associated set of endpoint addresses. A member's name is a string of type PIDName.

The returned Network Map MUST include all PIDs known to the ALTO Server.

7.7.2.1.6. Example

```
GET /networkmap HTTP/1.1
Host: alto.example.com
Accept: application/alto-networkmap+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-networkmap+json

{
  "meta" : {},
  "data" : {
    "map-vtag" : "1266506139",
    "map" : {
      "PID1" : {
        "ipv4" : [
          "192.0.2.0/24",
          "198.51.100.0/25"
        ]
      },
      "PID2" : {
        "ipv4" : [
          "198.51.100.128/25"
        ]
      },
      "PID3" : {
        "ipv4" : [
          "0.0.0.0/0"
        ],
        "ipv6" : [
          "::/0"
        ]
      }
    }
  }
}
```

7.7.2.2. Cost Map

The Cost Map resource lists the Path Cost for each pair of source/ destination PID defined by the ALTO Server for a given Cost Type and Cost Mode. This resource MUST be provided for at least the 'routingcost' Cost Type and 'numerical' Cost Mode.

Note that since this resource, an unfiltered Cost Map requested by an HTTP GET, does not indicate the desired Cost Mode or Cost Type as input parameters, an ALTO Server MUST indicate in an Information Resource Directory a unfiltered Cost Map Information Resource by specifying the capabilities (Section 7.7.2.2.4) with "cost-types" and "cost-modes" members each having a single element. This technique will allow an ALTO Client to determine a URI for an unfiltered Cost Map of the desired Cost Mode and Cost Type.

7.7.2.2.1. Media Type

The media type is "application/alto-costmap+json".

7.7.2.2.2. HTTP Method

This resource is requested using the HTTP GET method.

7.7.2.2.3. Input Parameters

None.

7.7.2.2.4. Capabilities

This resource may be defined for across multiple Cost Types and Cost Modes. The capabilities of an ALTO Server URI providing this resource are defined by a JSON Object of type CostMapCapability:

```
object {
  CostMode cost-modes<0..*>;
  CostType cost-types<0..*>;
} CostMapCapability;
```

with members:

cost-modes The Cost Modes (Section 5.1.2) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

cost-types The Cost Types (Section 5.1.1) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

An ALTO Server MUST support all of the Cost Types listed here for each of the listed Cost Modes. Note that an ALTO Server may provide multiple Cost Map Information Resources, each with different capabilities.

7.7.2.2.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceCostMap:

```
object DstCosts {
  JSONNumber [PIDName];
  ...
};

object {
  DstCosts [PIDName]<0..*>;
  ...
} CostMapData;

object {
  CostMode    cost-mode;
  CostType    cost-type;
  VersionTag  map-vtag;
  CostMapData map;
} InfoResourceCostMap;
```

with members:

cost-mode Cost Mode (Section 5.1.2) used in the Cost Map.

cost-type Cost Type (Section 5.1.1) used in the Cost Map.

map-vtag The Version Tag (Section 5.3) of the Network Map used to generate the Cost Map.

map The Cost Map data itself.

CostMapData is a JSON object with each member representing a single Source PID; the name for a member is the PIDName string identifying the corresponding Source PID. For each Source PID, a DstCosts object denotes the associated cost to a set of destination PIDs (Section 5.2); the name for each member in the object is the PIDName string identifying the corresponding Destination PID.

The returned Cost Map MUST include the Path Cost for each (Source PID, Destination PID) pair for which a Path Cost is defined. An ALTO Server MAY omit entries for which a Path Cost is not defined (e.g., both the Source and Destination PIDs contain addresses outside of the Network Provider's administrative domain).

7.7.2.2.6. Example

```
GET /costmap/num/routingcost HTTP/1.1
Host: alto.example.com
Accept: application/alto-costmap+json,application/alto-error+json
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : "numerical",
    "cost-type" : "routingcost",
    "map-vtag"  : "1266506139",
    "map" : {
      "PID1": { "PID1": 1,  "PID2": 5,  "PID3": 10 },
      "PID2": { "PID1": 5,  "PID2": 1,  "PID3": 15 },
      "PID3": { "PID1": 20, "PID2": 15  }
    }
  }
}
```

7.7.3. Map Filtering Service

The Map Filtering Service allows ALTO Clients to specify filtering criteria to return a subset of the full maps available in the Map Service.

7.7.3.1. Filtered Network Map

A Filtered Network Map is a Network Map Information Resource (Section 7.7.2.1) for which an ALTO Client may supply a list of PIDs to be included. A Filtered Network Map MAY be provided by an ALTO Server.

7.7.3.1.1. Media Type

See Section 7.7.2.1.1.

7.7.3.1.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.3.1.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-networkmapfilter+json", which is a JSON Object of type ReqFilteredNetworkMap, where:

```
object {
  PIDName pids<0..*>;
  AddressType address-types<0..*>;
} ReqFilteredNetworkMap;
```

with members:

pids Specifies list of PIDs to be included in the returned Filtered Network Map. If the list of PIDs is empty, the ALTO Server MUST interpret the list as if it contained a list of all currently-defined PIDs. The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

address-types Specifies list of address types to be included in the returned Filtered Network Map. If the list of address types is empty, the ALTO Server MUST interpret the list as if it contained a list of all address types known to the ALTO Server. The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

7.7.3.1.4. Capabilities

None.

7.7.3.1.5. Response

See Section 7.7.2.1.5 for the format.

The ALTO Server MUST only include PIDs in the response that were specified (implicitly or explicitly) in the request. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as if the PID did not appear in the input parameters. Similarly, the ALTO Server MUST only enumerate addresses within each PID that have types which were specified (implicitly or explicitly) in the request. If the input parameters contain an address type that is not currently known to the ALTO Server, the ALTO Server MUST behave as if the address type did not appear in the input parameters.

7.7.3.1.6. Example

```
POST /networkmap/filtered HTTP/1.1
Host: custom.alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-networkmapfilter+json
Accept: application/alto-networkmap+json,application/alto-error+json

{
  "pids": [ "PID1", "PID2" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-networkmap+json

{
  "meta" : {},
  "data" : {
    "map-vtag" : "1266506139",
    "map" : {
      "PID1" : {
        "ipv4" : [
          "192.0.2.0/24",
          "198.51.100.0/24"
        ]
      },
      "PID2" : {
        "ipv4": [
          "198.51.100.128/24"
        ]
      }
    }
  }
}
```

7.7.3.2. Filtered Cost Map

A Filtered Cost Map is a Cost Map Information Resource (Section 7.7.2.2) for which an ALTO Client may supply additional parameters limiting the scope of the resulting Cost Map. A Filtered Cost Map MAY be provided by an ALTO Server.

7.7.3.2.1. Media Type

See Section 7.7.2.2.1.

7.7.3.2.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.3.2.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-costmapfilter+json", which is a JSON Object of type ReqFilteredCostMap, where:

```
object {
  PIDName srcs<0..*>;
  PIDName dsts<0..*>;
} PIDFilter;

object {
  CostMode   cost-mode;
  CostType   cost-type;
  JSONString constraints<0..*>;   [OPTIONAL]
  PIDFilter  pids;                [OPTIONAL]
} ReqFilteredCostMap;
```

with members:

cost-type The Cost Type (Section 5.1.1) for the returned costs. This MUST be one of the supported Cost Types indicated in this resource's capabilities (Section 7.7.3.2.4).

cost-mode The Cost Mode (Section 5.1.2) for the returned costs. This MUST be one of the supported Cost Modes indicated in this resource's capabilities (Section 7.7.3.2.4).

constraints Defines a list of additional constraints on which elements of the Cost Map are returned. This parameter MUST NOT be specified if this resource's capabilities (Section 7.7.3.2.4) indicate that constraint support is not available. A constraint contains two entities separated by whitespace: (1) an operator either 'gt' for greater than or 'lt' for less than (2) a target numerical cost. The numerical cost is a number that MUST be defined in the same units as the Cost Type indicated by the cost-type parameter. ALTO Servers SHOULD use at least IEEE 754 double-

precision floating point [IEEE.754.2008] to store the numerical cost, and SHOULD perform internal computations using double-precision floating-point arithmetic. If multiple 'constraint' parameters are specified, they are interpreted as being related to each other with a logical AND.

pids A list of Source PIDs and a list of Destination PIDs for which Path Costs are to be returned. If a list is empty, the ALTO Server MUST interpret it as the full set of currently-defined PIDs. The ALTO Server MUST interpret entries appearing in a list multiple times as if they appeared only once. If the "pids" member is not present, both lists MUST be interpreted by the ALTO Server as containing the full set of currently-defined PIDs.

7.7.3.2.4. Capabilities

The URI providing this resource supports all capabilities documented in Section 7.7.2.2.4 (with identical semantics), plus additional capabilities. In particular, the capabilities are defined by a JSON object of type FilteredCostMapCapability:

```
object {
  CostMode cost-modes<0..*>;
  CostType cost-types<0..*>;
  JSONBool cost-constraints;
} FilteredCostMapCapability;
```

with members:

cost-modes See Section 7.7.2.2.4.

cost-types See Section 7.7.2.2.4.

cost-constraints If true, then the ALTO Server allows cost constraints to be included in requests to the corresponding URI. If not present, this member MUST be interpreted as if it specified false.

7.7.3.2.5. Response

See Section 7.7.2.2.5 for the format.

The returned Cost Map MUST NOT contain any source/destination pair that was not indicated (implicitly or explicitly) in the input parameters. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as

if the PID did not appear in the input parameters.

If any constraints are specified, Source/Destination pairs for which the Path Costs do not meet the constraints MUST NOT be included in the returned Cost Map. If no constraints were specified, then all Path Costs are assumed to meet the constraints.

Note that ALTO Clients should verify that the Version Tag included in the response is consistent with the Version Tag of the Network Map used to generate the request (if applicable). If it is not, the ALTO Client may wish to request an updated Network Map, identify changes, and consider requesting a new Filtered Cost Map.

7.7.3.2.6. Example

```
POST /costmap/filtered HTTP/1.1
Host: custom.alto.example.com
Content-Type: application/alto-costmapfilter+json
Accept: application/alto-costmap+json,application/alto-error+json

{
  "cost-mode" : "numerical",
  "cost-type" : "routingcost",
  "pids" : {
    "srcs" : [ "PID1" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-costmap+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : "numerical",
    "cost-type" : "routingcost",
    "map-vtag" : "1266506139",
    "map" : {
      "PID1": { "PID1": 0,  "PID2": 1,  "PID3": 2 }
    }
  }
}
```

7.7.4. Endpoint Property Service

The Endpoint Property Service provides information about Endpoint properties to ALTO Clients.

7.7.4.1. Endpoint Property

The Endpoint Property resource provides information about properties for individual endpoints. It MAY be provided by an ALTO Server. If an ALTO Server provides the Endpoint Property resource, it MUST provide and define at least the 'pid' property for each Endpoint.

7.7.4.1.1. Media Type

The media type is "application/alto-endpointprop+json".

7.7.4.1.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.4.1.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the data format of input parameteres with the media type "application/alto-endpointpropparams+json", which is a JSON Object of type ReqEndpointProp:

```
object {
  EndpointProperty  properties<1..*>;
  TypedEndpointAddr endpoints<1..*>;
} ReqEndpointProp;
```

with members:

properties List of endpoint properties to returned for each endpoint. Each specified property MUST be included in the list of supported properties indicated by this resource's capabilities (Section 7.7.4.1.4). The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

endpoints List of endpoint addresses for which the specified properties are to be returned. The ALTO Server MUST interpret entries appearing multiple times as if they appeared only once.

7.7.4.1.4. Capabilities

This resource may be defined across multiple types of endpoint properties. The capabilities of an ALTO Server URI providing Endpoint Properties are defined by a JSON Object of type EndpointPropertyCapability:

```
object {
  EndpointProperty prop-types<0..*>;
} EndpointPropertyCapability;
```

with members:

prop-types The Endpoint Property Types (Section 3.2.3) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

7.7.4.1.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceEndpointProperty, where:

```
object {
  JSONString [EndpointProperty];
  ...
} EndpointProps;

object {
  VersionTag     map-vtag;                      [OPTIONAL]
  EndpointProps  [TypedEndpointAddr]<0..*>;
  ...
} InfoResourceEndpointProperty;
```

InfoResourceEndpointProperty has one member for each endpoint indicated in the input parameters (with the name being the endpoint encoded as a TypedEndpointAddr). The requested properties for each endpoint are encoded in a corresponding EndpointProps object, which encodes one name/value pair for each requested property, where the property names are encoded as strings of type EndpointProperty and the property values encoded as JSON Strings.

The ALTO Server returns the value for each of the requested endpoint properties for each of the endpoints listed in the input parameters.

If the ALTO Server does not define a requested property for a particular endpoint, then it MUST omit it from the response for only that endpoint.

The ALTO Server MAY include the Version Tag (Section 5.3) of the Network Map used to generate the response (if desired and applicable) as the 'map-vtag' member in the response. If the 'pid' property is returned for any endpoints in the response, the 'map-vtag' member is REQUIRED instead of OPTIONAL.

7.7.4.1.6. Example

```
POST /endpointprop/lookup HTTP/1.1
Host: alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-endpointpropparams+json
Accept: application/alto-endpointprop+json,application/alto-error+json

{
  "properties" : [ "pid" ],
  "endpoints" : [ "ipv4:192.0.2.34", "ipv4:203.0.113.129" ]
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-endpointprop+json

{
  "meta" : {},
  "data": {
    "ipv4:192.0.2.34"    : { "pid": "PID1" },
    "ipv4:203.0.113.129" : { "pid": "PID3" }
  }
}
```

7.7.5. Endpoint Cost Service

The Endpoint Cost Service provides information about costs between individual endpoints.

In particular, this service allows lists of Endpoint prefixes (and addresses, as a special case) to be ranked (ordered) by an ALTO Server.

7.7.5.1. Endpoint Cost

The Endpoint Cost resource provides information about costs between individual endpoints. It MAY be provided by an ALTO Server.

It is important to note that although this resource allows an ALTO Server to reveal costs between individual endpoints, an ALTO Server is not required to do so. A simple alternative would be to compute the cost between two endpoints as the cost between the PIDs corresponding to the endpoints. See Section 12.1 for additional details.

7.7.5.1.1. Media Type

The media type is "application/alto-endpointcost+json".

7.7.5.1.2. HTTP Method

This resource is requested using the HTTP POST method.

7.7.5.1.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies input parameters with a data format indicated by media type "application/alto-endpointcostparams+json", which is a JSON Object of type ReqEndpointCostMap:

```
object {
  TypedEndpointAddr srcs<0..*>;    [OPTIONAL]
  TypedEndpointAddr dsts<1..*>;
} EndpointFilter;

object {
  CostMode          cost-mode;
  CostType          cost-type;
  JSONString        constraints;   [OPTIONAL]
  EndpointFilter    endpoints;
} ReqEndpointCostMap;
```

with members:

cost-mode The Cost Mode (Section 5.1.2) to use for returned costs. This MUST be one of the Cost Modes indicated in this resource's capabilities (Section 7.7.5.1.4).

cost-type The Cost Type (Section 5.1.1) to use for returned costs. This MUST be one of the Cost Types indicated in this resource's capabilities (Section 7.7.5.1.4).

constraints Defined equivalently to the "constraints" input parameter of a Filtered Cost Map (see Section 7.7.3.2).

endpoints A list of Source Endpoints and Destination Endpoints for which Path Costs are to be returned. If the list of Source Endpoints is empty (or not included), the ALTO Server MUST interpret it as if it contained the Endpoint Address corresponding to the client IP address from the incoming connection (see Section 10.3 for discussion and considerations regarding this mode). The list of destination Endpoints MUST NOT be empty. The ALTO Server MUST interpret entries appearing multiple times in a list as if they appeared only once.

7.7.5.1.4. Capabilities

See Section 7.7.3.2.4.

7.7.5.1.5. Response

The returned InfoResourceEntity object has "data" member equal to InfoResourceEndpointCostMap, where:

```
object EndpointDstCosts {
  JSONNumber [TypedEndpointAddr];
  ...
};

object {
  EndpointDstCosts [TypedEndpointAddr]<0..*>;
  ...
} EndpointCostMapData;

object {
  CostMode            cost-mode;
  CostType            cost-type;
  EndpointCostMapData map;
} InfoResourceEndpointCostMap;
```

InfoResourceEndpointCostMap has members:

cost-mode The Cost Mode used in the returned Cost Map.

cost-type The Cost Type used in the returned Cost Map.

map The Endpoint Cost Map data itself.

EndpointCostMapData is a JSON object with each member representing a single Source Endpoint specified in the input parameters; the name for a member is the TypedEndpointAddr string identifying the corresponding Source Endpoint. For each Source Endpoint, a EndpointDstCosts object denotes the associated cost to each Destination Endpoint specified in the input parameters; the name for each member in the object is the TypedEndpointAddr string identifying the corresponding Destination Endpoint. If the ALTO Server does not define a cost from a Source Endpoint to a particular Destination Endpoint, it MAY be omitted from the response.

7.7.5.1.6. Example

```
POST /endpointcost/lookup HTTP/1.1
Host: alto.example.com
Content-Length: [TODO]
Content-Type: application/alto-endpointcostparams+json
Accept: application/alto-endpointcost+json,application/alto-error+json

{
  "cost-mode" : "ordinal",
  "cost-type" : "routingcost",
  "endpoints" : {
    "srcs": [ "ipv4:192.0.2.2" ],
    "dsts": [
      "ipv4:192.0.2.89",
      "ipv4:198.51.100.34",
      "ipv4:203.0.113.45"
    ]
  }
}
```

```
HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-endpointcost+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : "ordinal",
    "cost-type" : "routingcost",
    "map" : {
      "ipv4:192.0.2.2": {
        "ipv4:192.0.2.89"    : 1,
        "ipv4:198.51.100.34" : 2,
        "ipv4:203.0.113.45"  : 3
      }
    }
  }
}
```

8. Redistributable Responses

This section defines how an ALTO Server enables certain Information Resources to be redistributed by ALTO Clients. Concepts are first

introduced, followed by the protocol specification.

8.1. Concepts

8.1.1. Service ID

The Service ID is a UUID that identifies a set of ALTO Servers that would provide semantically-identical Information Resources for any request for any ALTO Client. Each ALTO Server within such a set is configured with an identical Service ID.

If a pair of ALTO Servers would provide an identical Information Resource (same information sources, configuration, internal computations, update timescales, etc) in response to any particular ALTO Client request, then the pair of ALTO Servers MAY have the same Service ID. If this condition is not true, the pair of ALTO Servers MUST have a different Service ID.

8.1.1.1. Rationale

For scalability and fault tolerance, multiple ALTO Servers may be deployed to serve equivalent ALTO Information. In such a scenario, Information Resources from any such redundant server should be seen as equivalent for the purposes of redistribution. For example, if two ALTO Servers A and B are deployed by the service provider to distribute equivalent ALTO Information, then clients contacting Server A should be able to redistribute Information Resources to clients contacting Server B.

To accomplish this behavior, ALTO Clients must be able to determine that Server A and Server B serve identical ALTO Information. One technique would be to rely on the ALTO Server's DNS name. However, such an approach would mandate that all ALTO Servers resolved by a particular DNS name would need to provide equivalent ALTO information, which may be unnecessarily restrictive. Another technique would be to rely on the server's IP address. However, this suffers similar problems as the DNS name in deployment scenarios using IP Anycast.

To avoid such restrictions, the ALTO Protocol allows an ALTO Service Provider to explicitly denote ALTO Servers that provide equivalent ALTO Information by giving them identical Service IDs. Service IDs decouple the identification of equivalent ALTO Servers from the discovery process.

8.1.1.2. Server Information Resource

If an ALTO Server generates redistributable responses, the Server Information resource's 'service-id' field MUST be set to the ALTO Server's Service ID.

8.1.1.3. Configuration

To help prevent ALTO Servers from mistakenly claiming to distribute equivalent ALTO Information, ALTO Server implementations SHOULD by default generate a new UUID at installation time or startup if one has not explicitly been configured.

8.1.2. Expiration Time

Information Resources marked as redistributable should indicate a time after which the information is considered stale and should be refreshed from the ALTO Server (or possibly another ALTO Client).

If an expiration time is present, the ALTO Server SHOULD ensure that it is reasonably consistent with the expiration time that would be computed by HTTP header fields. This specification makes no recommendation on which expiration time takes precedence, but implementers should be cognizant that HTTP intermediaries will obey only the HTTP header fields.

8.1.3. Signature

Information Resources marked as redistributable include a signature used to assert that the ALTO Server Provider generated the ALTO Information.

8.1.3.1. Rationale

Verification of the signature requires the ALTO Client to retrieve the ALTO Server's public key. To reduce requirements on the underlying transport (i.e., requiring SSL/TLS), an ALTO Client retrieves the public key as part of an X.509 certificate from the ALTO Server's Server Information resource.

8.1.3.2. Certificates

8.1.3.2.1. Local Certificate

The ALTO Server's public key is encoded within an X.509 certificate. The corresponding private key MUST be used to sign redistributable responses. This certificate is termed the Local Certificate for an ALTO Server.

8.1.3.2.2. Certificate Chain

To ease key provisioning, the ALTO Protocol is designed such that each ALTO Server with an identical Service ID may have a unique private key (and hence certificate).

The ALTO Service Provider may configure a certificate chain at each such ALTO Server. The Local Certificate for a single ALTO Server is the bottom-most certificate in the chain. The Certificate Chains of each ALTO Server with an identical Service ID MUST share a common Root Certificate.

Note that there are two simple deployment scenarios:

- One-Level Certificate Chain (Local Certificate Only): In this deployment scenario, each ALTO Server with an identical Service ID may provisioned with an identical Local Certificate.

- Two-Level Certificate Chain: In this deployment scenario, a Root Certificate is maintained for a set of ALTO Servers with the same Service ID. A unique Local Certificate signed by this CA is provisioned to each ALTO Server.

There are advantages to using a Certificate Chain instead of deploying the same Local Certificate to each ALTO Server. Specifically, it avoids storage of the CA's private key at ALTO Servers. It is possible to revoke and re-issue a key to a single ALTO Server.

8.1.3.2.3. Server Information Resource

If an ALTO Server generates redistributable responses, the Server Information resource's 'certificates' field MUST be populated with the ALTO Server's full certificate chain. The first element MUST be the ALTO Server's Local Certificate, followed by the remaining Certificate Chain in ascending order to the Root Certificate.

8.1.3.3. Signature Verification

ALTO Clients SHOULD verify the signature on any ALTO information received via redistribution before adjusting application behavior based on it.

An ALTO Client SHOULD cache its ALTO Server's Service ID and corresponding Certificate Chain included in the Server Information resource. Recall that the last certificate in this chain is the Root Certificate. The retrieval of the Service ID and certificates SHOULD be secured using HTTPS with proper validation of the server endpoint

of the SSL/TLS connection [RFC6125].

An Information Resource received via redistribution from Service ID S is declared valid if an ALTO Client can construct a transitive certificate chain from the certificate (public key) used to sign the Information Resource to the Root Certificate corresponding to Service ID S obtained by the ALTO Client in a Server Information resource.

To properly construct the chain and complete this validation, an ALTO Client may need to request additional certificates from other ALTO Clients. A simple mechanism is to request the certificate chain from the ALTO Client that received the Information Resource. Note that these additional received certificates may be cached locally by an ALTO Client.

ALTO Clients SHOULD verify Information Resources received via redistribution.

8.1.3.4. Redistribution by ALTO Clients

ALTO Clients SHOULD pass the ALTO Server Certificate, Signature, and Signature Algorithm along with the Information Resource. The mechanism for redistributing such information is not specified by the ALTO Protocol, but one possibility is to add additional messages or fields to the application's native protocol.

8.2. Protocol

An ALTO Server MAY indicate that a response is suitable for redistribution by including the "redistribution" member in the RspMetaData JSON object of an Information Resource. This additional member, called the Response Redistribution Descriptor, has type InfoResourceRedistDesc:

```
object {
  JSONString service-id;
  JSONString request-uri;
  JSONValue  request-body;
  JSONString media-type;
  JSONString expires;
} InfoResourceRedistDesc;
```

The fields encoded in the Response Redistribution Descriptor allows an ALTO Client receiving redistributed ALTO Information to understand the context of the query (the ALTO Service generating the response and any input parameters) and to interpret the results.

Information about ALTO Client performing the request and any HTTP

Headers passed in the request are not included in the Response Redistribution Descriptor. If any such information or headers influence the response generated by the ALTO Server, the response SHOULD NOT be indicated as redistributable.

8.2.1. Response Redistribution Descriptor Fields

This section defines the fields of the Response Redistribution Descriptor.

8.2.1.1. Service ID

The 'service-id' member is REQUIRED and MUST have a value equal to the ALTO Server's Service ID.

8.2.1.2. Request URI

The 'request-uri' member is REQUIRED and MUST specify the HTTP Request-URI that was passed in the HTTP request.

8.2.1.3. Request Body

If the HTTP request's entity body was non-empty, the 'request-body' member MUST specify full JSON value passed in the HTTP request's entity body (note that whitespace may differ, as long as the JSON Value is identical). If the HTTP request was empty, then the 'request-body' MUST NOT be included.

8.2.1.4. Response Media Type

The 'media-type' member is REQUIRED and MUST specify the same HTTP Content-Type that is used in the HTTP response.

8.2.1.5. Expiration Time

The 'expires' element is RECOMMENDED and, if present, MUST specify a time in UTC formatted according to [RFC3339].

8.2.2. Signature

The Hash Algorithm, Signature Algorithm, and Signature are included as either HTTP Headers or Trailers. Headers may be useful if Information Resources are pre-generated, while Trailers may be useful if Information Resources are dynamically generated (e.g., to avoid buffering large responses in memory while the hash value is computed).

The following HTTP Headers (the ALTO Server MAY specify them as HTTP

Trailers instead) MUST be used to encode the Signature parameters for redistributable Information Resources:

```
ALTO-HashAlgorithm: <HashAlgorithm>
ALTO-SignatureAlgorithm: <SignatureAlgorithm>
ALTO-SignatureDigest: <Signature>
```

where <HashAlgorithm> and <SignatureAlgorithm> are an integer values from the IANA TLS HashAlgorithm and SignatureAlgorithm registries, and <Signature> is the corresponding Base64-encoded signature.

9. Use Cases

The sections below depict typical use cases.

9.1. 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.

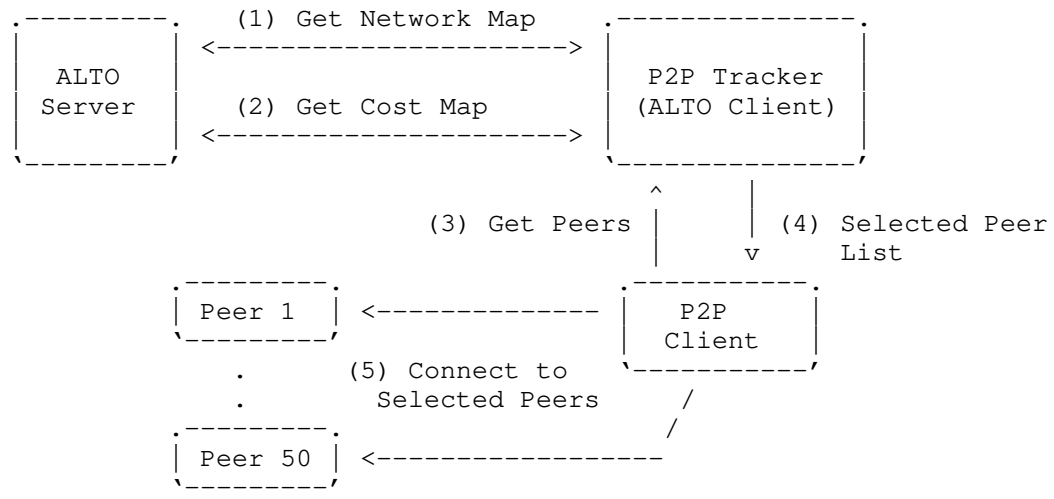


Figure 4: ALTO Client Embedded in P2P Tracker

Figure 4 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 Network Map 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.

9.2. 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.

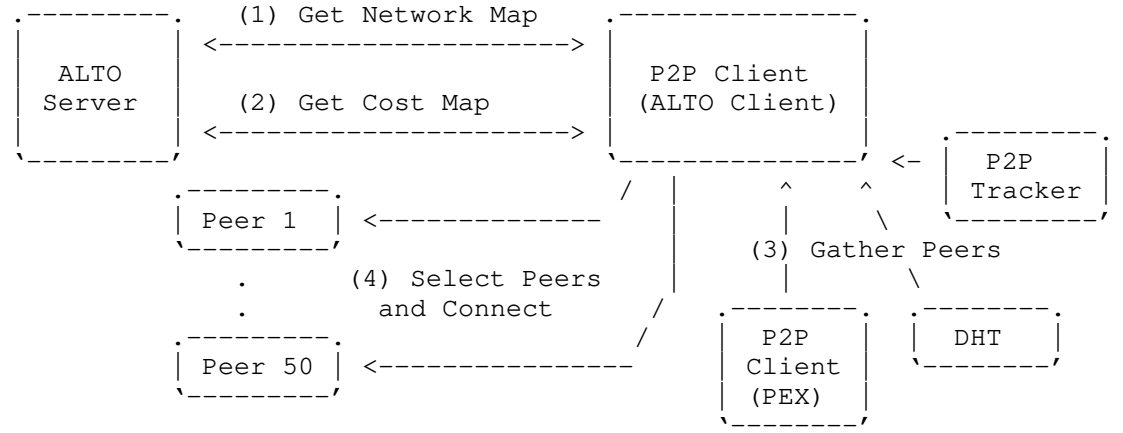


Figure 5: ALTO Client Embedded in P2P Client

Figure 5 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.

9.3. 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.

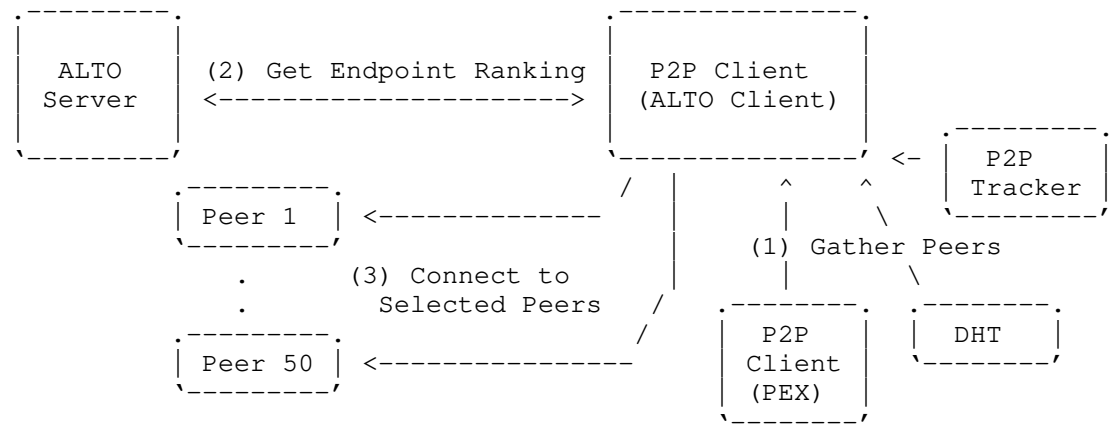


Figure 6: ALTO Client Embedded in P2P Client: Ranking

Figure 6 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 the ALTO Server's Ranking Service, including discovered peers as the set of Destination Endpoints, and indicates the 'ordinal' Cost Mode. The response indicates the ranking of the candidate peers.

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

10. Discussions

10.1. Discovery

The discovery mechanism by which an ALTO Client locates an appropriate ALTO Server is out of scope for this document. This document assumes that an ALTO Client can discover an appropriate ALTO Server. Once it has done so, the ALTO Client may use the Information Resource Directory (see Section 7.6) to locate an Information Resource with the desired ALTO Information.

10.2. Hosts with Multiple Endpoint Addresses

In practical deployments, especially during the transition from IPv4 to IPv6, a particular host may be reachable using multiple addresses. Furthermore, the particular network path followed when sending packets to the host may differ based on the address that is used. Network providers may prefer one path over another (e.g., one path my have a NAT64 middlebox). An additional consideration may be how to handle private address spaces (e.g., behind carrier-grade NATs).

To support such behavior, this document allows multiple types of endpoint addresses. In supporting multiple address types, the ALTO Protocol also allows ALTO Service Provider the flexibility to indicate preferences for paths from an endpoint address of one type to an endpoint address of a different type. Note that in general, the path through the network may differ dependent on the types of addresses that are used.

Note that there are limitations as to what information ALTO can provide in this regard. In particular, a particular ALTO Service provider may not be able to determine if connectivity with a particular endhost will succeed over IPv4 or IPv6, as this may depend upon information unknown to the ISP such as particular application implementations.

10.3. Network Address Translation Considerations

At this day and age of NAT v4<->v4, v4<->v6 [RFC6144], and possibly v6<->v6[I-D.mrw-nat66], 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 where the source network location is computed by the ALTO Server (i.e., the the Endpoint Cost Service) 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 [BitTorrent]) operate.

The ALTO client SHOULD use the Session Traversal Utilities for NAT (STUN) [RFC5389] to determine a public IP address to use as a source Endpoint address. If using this method, the host MUST use 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.

10.4. 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, the Network Map query in the Map Filtering Service 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.

10.5. 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. This specification focuses on network location and grouping, but the protocol may be extended to handle other Endpoint properties.

11. IANA Considerations

11.1. application/alto-* Media Types

This document requests the registration of multiple media types, listed in Table 2.

Type	Subtype	Specification
application	alto-directory+json	Section 7.6
application	alto-serverinfo+json	Section 7.7.1.1
application	alto-networkmap+json	Section 7.7.2.1
application	alto-networkmapfilter+json	Section 7.7.3.1
application	alto-costmap+json	Section 7.7.2.2
application	alto-costmapfilter+json	Section 7.7.3.2
application	alto-endpointprop+json	Section 7.7.4.1
application	alto-endpointpropparams+json	Section 7.7.4.1
application	alto-endpointcost+json	Section 7.7.5.1
application	alto-endpointcostparams+json	Section 7.7.5.1
application	alto-error+json	Section 7.4

Table 2: ALTO Protocol Media Types

Type name: application

Subtype name: This documents requests the registration of multiple subtypes, as listed in Table 2.

Required parameters: n/a

Optional parameters: n/a

Encoding considerations: Encoding considerations are identical to those specified for the 'application/json' media type. See [RFC4627].

Security considerations: Security considerations relating to the generation and consumption of ALTO protocol messages are discussed in Section 12.

Interoperability considerations: This document specifies format of conforming messages and the interpretation thereof.

Published specification: This document is the specification for these media types; see Table 2for the section documenting each media type.

Applications that use this media type: ALTO Servers and ALTO Clients either standalone or embedded within other applications.

Additional information:

Magic number(s): n/a

File extension(s): This document uses the mime type to refer to protocol messages and thus does not require a file extension.

Macintosh file type code(s): n/a

Person & email address to contact for further information: See "Authors' Addresses" section.

Intended usage: COMMON

Restrictions on usage: n/a

Author: See "Authors' Addresses" section.

Change controller: See "Authors' Addresses" section.

11.2. ALTO Cost Type Registry

This document requests the creation of an ALTO Cost Type registry to be maintained by IANA.

This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO Cost Types. Second, it provides references to particular semantics of allocated Cost Types to be applied by both ALTO Servers and applications utilizing ALTO Clients.

New ALTO Cost Types are assigned after Expert Review [RFC5226]. The Expert Reviewer will generally consult the ALTO Working Group or its successor. Expert Review is used to ensure that proper documentation regarding ALTO Cost Type semantics and security considerations has been provided. The provided documentation should be detailed enough to provide guidance to both ALTO Service Providers and applications utilizing ALTO Clients as to how values of the registered ALTO Cost Type should be interpreted. Updates and deletions of ALTO Cost Types follow the same procedure.

Registered ALTO Cost Type identifiers MUST conform to the syntatical requirements specified in Section 7.5.5. Identifiers are to be recorded and displayed as ASCII strings.

Identifiers prefixed with 'priv:' are reserved for Private Use. Identifiers prefixed with 'exp:' are reserved for Experimental use.

Requests to add a new value to the registry MUST include the

following information:

- Identifier: The name of the desired ALTO Cost Type.

- Intended Semantics: ALTO Costs carry with them semantics to guide their usage by ALTO Clients. For example, if a value refers to a measurement, the measurement units must be documented. For proper implementation of the ordinal Cost Mode (e.g., by a third-party service), it should be documented whether higher or lower values of the cost are more preferred.

- Security Considerations: ALTO Costs expose information to ALTO Clients. As such, proper usage of a particular Cost Type may require certain information to be exposed by an ALTO Service Provider. Since network information is frequently regarded as proprietary or confidential, ALTO Service Providers should be made aware of the security ramifications related to usage of a Cost Type.

This specification requests registration of the identifier 'routingcost'. Semantics for the this Cost Type are documented in Section 5.1.1.1, and security considerations are documented in Section 12.1.

11.3. ALTO Endpoint Property Registry

This document requests the creation of an ALTO Endpoint Property registry to be maintained by IANA.

This registry serves two purposes. First, it ensures uniqueness of identifiers referring to ALTO Endpoint Properties. Second, it provides references to particular semantics of allocated Endpoint Properties to be applied by both ALTO Servers and applications utilizing ALTO Clients.

New ALTO Endpoint Properties are assigned after Expert Review [RFC5226]. The Expert Reviewer will generally consult the ALTO Working Group or its successor. Expert Review is used to ensure that proper documentation regarding ALTO Endpoint Property semantics and security considerations has been provided. The provided documentation should be detailed enough to provide guidance to both ALTO Service Providers and applications utilizing ALTO Clients as to how values of the registered ALTO Endpoint Properties should be interpreted. Updates and deletions of ALTO Endpoint Properties follow the same procedure.

Registered ALTO Endpoint Property identifiers MUST conform to the syntatical requirements specified in Section 7.5.6. Identifiers are

to be recorded and displayed as ASCII strings.

Identifiers prefixed with 'priv:' are reserved for Private Use. Identifiers prefixed with 'exp:' are reserved for Experimental use.

Requests to add a new value to the registry MUST include the following information:

- Identifier: The name of the desired ALTO Endpoint Property.
- Intended Semantics: ALTO Endpoint Properties carry with them semantics to guide their usage by ALTO Clients. For example, if a value refers to a measurement, the measurement units must be documented. For proper implementation of the ordinal Cost Mode (e.g., by a third-party service), it should be documented whether higher or lower values of the cost are more preferred.
- Security Considerations: ALTO Endpoint Properties expose information to ALTO Clients. As such, proper usage of a particular Endpoint Properties may require certain information to be exposed by an ALTO Service Provider. Since network information is frequently regarded as proprietary or confidential, ALTO Service Providers should be made aware of the security ramifications related to usage of an Endpoint Property.

This specification requests registration of the identifier 'pid'. Semantics for the this Endpoint Property are documented in Section 4.1, and security considerations are documented in Section 12.1.

12. Security Considerations

12.1. Privacy Considerations for 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. On the one hand, providing overly fine-grained information may make it easier for attackers to infer network topology. In particular, attackers may try to infer details regarding ISPs' operational policies or inter-ISP business relationships by intentionally posting a multitude of selective queries to an ALTO server and analyzing the responses. Such sophisticated attacks may reveal more information than an ISP hosting an ALTO server intends to disclose. On the other hand, revealing overly coarse-grained information may not provide benefits to network efficiency or performance improvements to ALTO Clients.

12.2. ALTO Clients

Applications using the information must be cognizant of the possibility that the information is malformed or incorrect. Even if an ALTO Server has been properly authenticated by the ALTO Client, the information provided may be malicious because the ALTO Server and its credentials have been compromised (e.g., through malware). Other considerations (e.g., relating to application performance) can be found in Section 6 of [RFC5693].

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.

In addition, ALTO clients should be cautious not to unintentionally or indirectly disclose the resource identifier (of which they try to improve the retrieval through ALTO-guidance), e.g., the name/ identifier of a certain video stream in P2P live streaming, to the ALTO server. Note that the ALTO Protocol specified in this document does not explicitly reveal any resource identifier to the ALTO Server. However, for instance, depending on the popularity or other specifics (such as language) of the resource, an ALTO server could potentially deduce information about the desired resource from information such as the Network Locations the client sends as part of its request to the server.

12.3. Authentication, Integrity Protection, and Encryption

SSL/TLS can provide encryption of transmitted messages as well as authentication of the ALTO Client and Server. HTTP Basic or Digest authentication can provide authentication of the client (combined with SSL/TLS, it can additionally provide encryption and authentication of the server).

An ALTO Server may optionally use authentication (and potentially encryption) to protect ALTO information it provides. This can be achieved by digitally signing a hash of the ALTO information itself and attaching the signature to the ALTO information. There may be special use cases where encryption of ALTO information is desirable. In many cases, however, information sent out by an ALTO Server may be regarded as non-confidential information.

ISPs should be cognizant that encryption only protects ALTO information until it is decrypted by the intended ALTO Client. Digital Rights Management (DRM) techniques and legal agreements

protecting ALTO information are outside of the scope of this document.

12.4. 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 generated by an appropriate 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 validate every set of information. Contacting an ALTO server for information validation would also undermine the intended effect of redistribution and is therefore not desirable.

Note that the redistribution scheme must additionally handle details such as ensuring ALTO Clients retrieve ALTO information from the correct ALTO Server. See [I-D.gu-alto-redistribution] for further discussion. Details of a particular redistribution scheme are outside the scope of this document.

To fulfill these requirements, ALTO Information meant to be redistributable contains a digital signature which includes a hash of the ALTO information signed by the ALTO Server with its private key. The corresponding public key is included in the Server Information resource Section 7.7.1.1, along with the certificate chain to a Root Certificate generated by the ALTO Service Provider. To prevent man-in-the-middle attacks, an ALTO Client SHOULD perform the Server Information resource request over SSL/TLS and verify the server identity according to [RFC6125].

The signature verification algorithm is detailed in Section 8.1.3.3.

12.5. Denial of Service

ISPs should be cognizant of the workload at the ALTO Server generated by certain ALTO Queries, such as certain queries to the Map Filtering Service and Ranking Service. In particular, queries which can be generated with low effort but result in expensive workloads at the ALTO Server could be exploited for Denial-of-Service attacks. For instance, a simple ALTO query with n Source Network Locations and m Destination Network Locations can be generated fairly easily but results in the computation of n*m Path Costs between pairs by the ALTO Server (see Section 5.2). One way to limit Denial-of-Service attacks is to employ access control to the ALTO server. Another possible mechanism for an ALTO Server to protect itself against a multitude of computationally expensive bogus requests is to demand

that each ALTO Client to solve a computational puzzle first before allocating resources for answering a request (see, e.g., [I-D.jennings-sip-hashcash]). The current specification does not use such computational puzzles, and discussion regarding tradeoffs of such an approach would be needed before including such a technique in the ALTO Protocol.

ISPs should also leverage the fact that the the Map Service allows ALTO Servers to pre-generate maps that can be useful to many ALTO Clients.

12.6. ALTO Server Access Control

In order to limit access to an ALTO server (e.g., for an ISP to only allow its users to access its ALTO server, or to prevent Denial-of-Service attacks by arbitrary hosts from the Internet), an ALTO server may employ access control policies. Depending on the use-case and scenario, an ALTO server may restrict access to its services more strictly or rather openly (see [I-D.stiemerling-alto-deployments] for a more detailed discussion on this issue).

13. References

13.1. Normative References

[IEEE.754.2008] Institute of Electrical and Electronics Engineers, "Standard for Binary Floating-Point Arithmetic", IEEE Standard 754, August 2008.

[RFC2046] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", RFC 2046, November 1996.

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

[RFC2616] 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.

[RFC3339] Klyne, G., Ed. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, July 2002.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.

[RFC4632] Fuller, V. and T. Li, "Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan", BCP 122, RFC 4632, August 2006.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, August 2010.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

13.2. Informative References

[BitTorrent]
"Bittorrent Protocol Specification v1.0", <http://wiki.theory.org/BitTorrentSpecification>.

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

[I-D.gu-alto-redistribution]
Yingjie, G., Alimi, R., and R. Even, "ALTO Information Redistribution", draft-gu-alto-redistribution-03 (work in progress), July 2010.

[I-D.ietf-alto-reqs]
Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO)

Requirements", draft-ietf-alto-reqs-08 (work in progress), March 2011.

[I-D.ietf-alto-server-discovery]
Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M., and S. Yongchao, "ALTO Server Discovery", draft-ietf-alto-server-discovery-02 (work in progress), September 2011.

[I-D.jennings-sip-hashcash]
Jennings, C., "Computational Puzzles for SPAM Reduction in SIP", draft-jennings-sip-hashcash-06 (work in progress), July 2007.

[I-D.mrw-nat66]
Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", draft-mrw-nat66-16 (work in progress), April 2011.

[I-D.p4p-framework]
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.

[I-D.saumitra-alto-multi-ps]
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.

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

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

[I-D.stiemerling-alto-deployments]
Stiemerling, M. and S. Kiesel, "ALTO Deployment Considerations", draft-stiemerling-alto-deployments-06 (work in progress), January 2011.

[I-D.wang-alto-p4p-specification]
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.

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

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

[RFC6144] Baker, F., Li, X., Bao, C., and K. Yin, "Framework for IPv4/IPv6 Translation", RFC 6144, April 2011.

Appendix A. Acknowledgments

Thank you to Jan Seedorf for contributions to the Security Considerations section. We would like to thank Yingjie Gu and Roni Even for helpful input and design concerning ALTO Information redistribution.

We would like to thank the following people whose input and involvement was indispensable in achieving this merged proposal:

Obi Akonjang (DT Labs/TU Berlin),

Saumitra M. Das (Qualcomm Inc.),

Syon Ding (China Telecom),

Doug Pasko (Verizon),

Laird Popkin (Pando Networks),

Satish Raghunath (Juniper Networks),

Albert Tian (Ericsson/Redback),

Yu-Shun Wang (Microsoft),

David Zhang (PPLive),

Yunfei Zhang (China Mobile).

We would also 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).

Appendix B. Authors

[[CmtAuthors: RFC Editor: Please move information in this section to the Authors' Addresses section at publication time.]]

Stefano Previdi
Cisco

Email: sprevidi@cisco.com

Stanislav Shalunov
BitTorrent

Email: shalunov@bittorrent.com

Richard Woundy
Comcast

Richard_Woundy@cable.comcast.com

Authors' Addresses

Richard Alimi (editor)
Google
1600 Amphitheatre Parkway
Mountain View CA
USA

Email: ralimi@google.com

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

Email: rpenno@juniper.net

Y. Richard Yang (editor)
Yale University
51 Prospect St
New Haven CT
USA

Email: yry@cs.yale.edu

ALTO
Internet-Draft
Intended status: Standards Track
Expires: January 12, 2012

S. Kiesel
University of Stuttgart
M. Stiemerling
NEC Europe Ltd.
N. Schwan
M. Scharf
Alcatel-Lucent Bell Labs
H. Song
Huawei

July 11, 2011

ALTO Server Discovery
draft-ietf-alto-server-discovery-01

Abstract

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource.

Entities seeking guidance need to discover and possibly select an ALTO server to ask. This is called ALTO server discovery. This memo describes an ALTO server discovery mechanism based on several alternative mechanisms that are applicable in a diverse set of ALTO deployments.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on January 12, 2012.

Copyright Notice

Table of Contents

1. Introduction

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications, which have to select one or several hosts from a set of candidates, that are able to provide a desired resource [RFC5693]. The requirements for ALTO are itemized in [I-D.ietf-alto-reqs]. ALTO is realized by a client-server protocol. ALTO clients send queries to ALTO servers, in order to solicit guidance.

ALTO clients have to discover suitable ALTO servers. Therefore the output of the herein defined ALTO discovery procedure tells the ALTO client which ALTO servers to send the queries to. The ALTO discovery procedure, as part of the ALTO client, can be embedded in the resource consumer, which will eventually access the desired resource. As an alternative, they can be embedded in a resource directory, which assists resource consumers in finding appropriate resource providers. In some specific peer-to-peer application protocols these resource directories are called "trackers". Finally the ALTO server discovery procedure can be embedded in the resource consumer, whereas the ALTO client is embedded in the resource directory. ALTO queries, which are issued by a resource directory on behalf of a resource consumer, are referred to as third-party ALTO queries. The various possibilities to place ALTO servers and the placement of ALTO clients is discussed in [I-D.ietf-alto-deployments].

No matter where ALTO server and client are located, clients have to first find out if there is an ALTO server deployed that is in charge for them, and second they have to get the contact information of that server, i.e., the IP address, port number, and probably transport protocol (which defaults to TCP for the ALTO protocol specification [I-D.ietf-alto-protocol]).

The goal of this memo is to propose a uniform mechanism for all types of ALTO client deployments that is implementable and deployable at a fast pace, i.e., without creating other deployment dependencies for ALTO. We propose a schema which employs the UNAPTR mechanism [RFC4848] to determine the URI of the ALTO server and where mutliple input methods to the UNAPTR process can be used.

Comments and discussions about this memo should be directed to the ALTO working group: alto@ietf.org.

1.1. History

[RFC editor's note: Please remove this section before publication.]

This document represents a merge of features from two previous

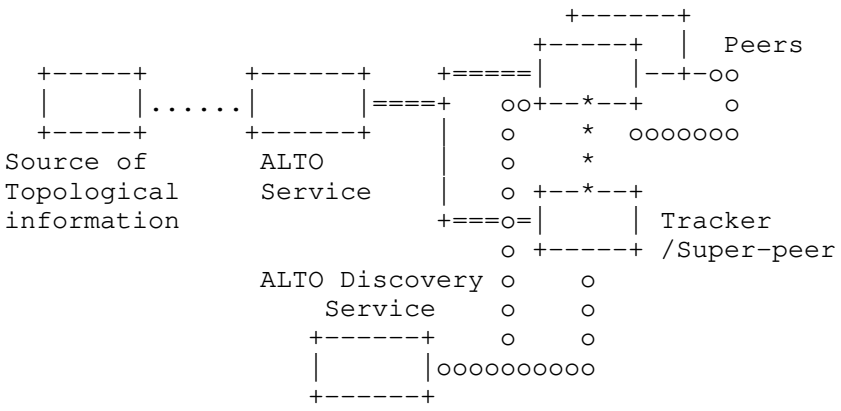
drafts:

- draft-kiesel-alto-3pdisc-04

- draft-song-alto-server-discovery-03

1.2. Discovery Scenarios

Figure 1 below shows an overview on the different entities of a generic ALTO framework. The ALTO Server discovery mechanism is used by the peer-to-peer (P2P) application in order retrieve the point of contact of the ALTO Service.



Legend:

=== ALTO query protocol
ooo ALTO service discovery protocol
*** Application protocol (out of scope)
... Provisioning or initialization (out of scope)

Figure 1: ALTO Discovery Overview

Hereby the ALTO service discovery scenarios are classified into two types: one is the ALTO server discovery by the resource consumer, and the other is the ALTO server discovery by a third party, such as application trackers. Before the specification of the discovery mechanism the following section illustrates and discusses both scenarios.

1.2.1. ALTO Server Discovery by Resource Consumers

The ALTO service discovery in some scenarios needs to be performed by the resource consumer itself. In particular in P2P applications without a tracker like DHTs and other conventional client/server applications.

In addition also P2P application which are tracker based may embed the ALTO client into the resource consumer to allow peers a selection of peers after retrieving the peer list from the application tracker. Another option is that the resource consumer peer sends its ALTO server address information to the application tracker or any other third party entity, which in turn will contact the specific ALTO server in order to retrieve ALTO guidance on behalf of the resource consumer.

The following figure illustrates this scenario, showing the relationship between the different entities as discussed before.

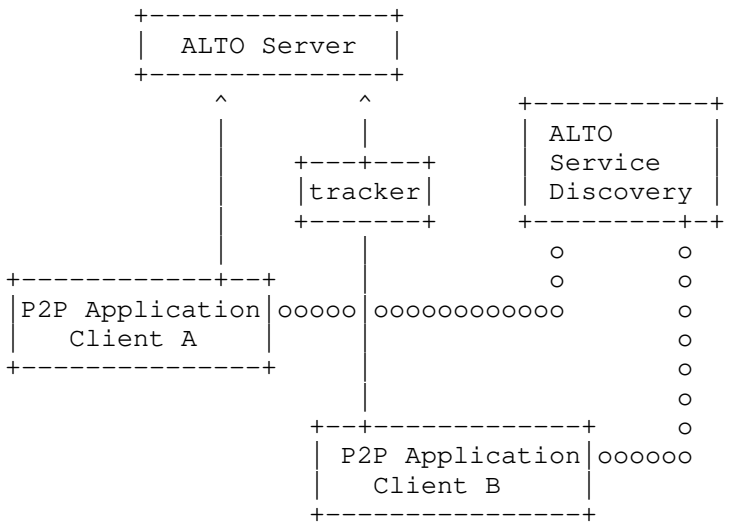


Figure 2: Resource Consumer ALTO Server Discovery (Example)

1.2.2. ALTO Server Discovery by a Third Party

Some P2P applications have trackers, and these applications might not need to have their clients looking for the ALTO server guidance. In these scenarios trackers query the ALTO servers for guidance themselves, and then return the final ranked result to the application clients. However, application clients are distributed among different network operators and autonomous systems. Trackers

thus need to find different ALTO servers for the clients located in different operator networks or autonomous systems. In such scenarios the discovery is thus not performed by the resource consumer, but a third party entity on behalf of the resource consumer.

Figure 3 shows an example for a third party ALTO server discovery. For Client1 (1), the tracker has not cached yet the mapping between Client1's network operator and its ALTO server address, so it uses the ALTO Discovery Service to determine the address of the ALTO server in that operator's domain (2). Then the tracker interacts with ALTO Server1 (3)(4) on behalf of Client1 (to get the network map and cost map), finally, the ranked list is sent back to Client1 (5). For Client2, the tracker has cached the mapping between Client2's network operator and ALTO Server2's address, so it does not need to perform the discovery process (which are the labels (a),(b), (c), and (d)). If the application tracker already has the network map and cost map from ALTO Server2, then it does not need to query the ALTO Server for network map and cost map frequently.

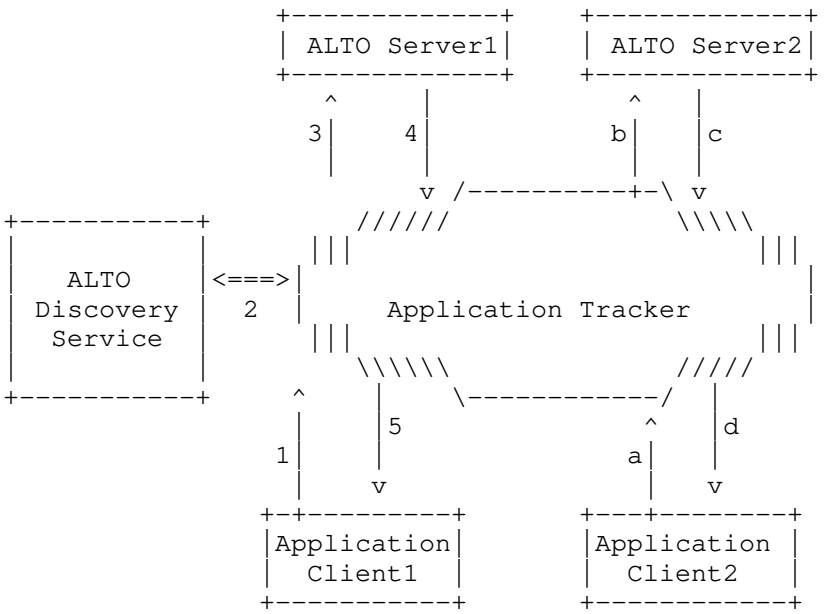


Figure 3: Third Party ALTO Server Discovery (Example)

1.3. Pre-Conditions

The whole document assumes certain pre-conditions, such as:

- The ALTO server discovery procedure is executed on a per IP address base. Multiple IP addresses per interface or multiple IP addresses assigned to different IP interfaces require to repeat the procedure for every IP address. It may be fine to group IP addresses according their domain suffixes and to perfom the procedure for such a group. However, this is out of scope of this document.[Editor's note: this may relate to the work of the MIF WG]

- The ALTO server discovery procedure is executed on a per IP family base, i.e., seperate for IPv4 and IPv6. It is up to the ALTO client to decide which of the possible multiple results of different IP address families to use. The choice of whether to use IPv4 or IPv6 is out of scope of this document.

- A change of the IP address at an interface invalidates the result of the ALTO server discovery procedure. For instance, if the IP address assigned to a mobile host changes due to host mobility, it is required to run the ALTO server discovery procedure for the new IP address without relying on earlier gained information.

2. Protocol Overview

We define multiple alternatives to discover the IP address of the ALTO server, as there are a number of ways possible how such information can be provided to the ALTO client. The choice of method is up to the local network deployment. For instance, there can be deployments where the ALTO server in charge for ALTO client is provisioned by the network operator and communicated to the ALTO client's host via a DHCP option, while in other deployments no such means may exist. It should be noted that there is no silver bullet solution to the ALTO server discovery, as there too many deployment scenarios in the server discovery space.

The following figure illustrates the different protocols that are used to find the URI of a suitable ALTO server.

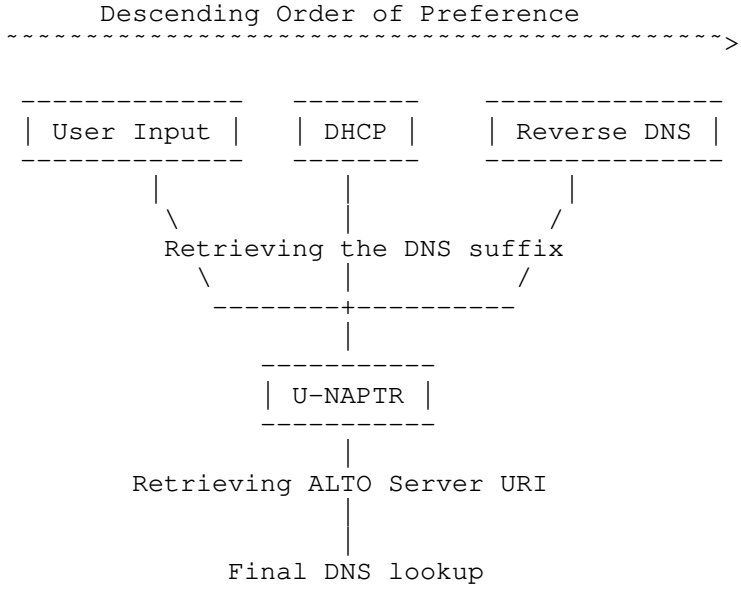


Figure 4: Protocol Overview

Figure 4 illustrates the U-NAPTR based resolution process to retrieve the ALTO Server URL. As a precondition for resolution the U-NAPTR process needs the right domain name as input. This domain name is determined by the IP address of the client and the DNS suffix of the access network where the client is registered in. In order to retrieve the DNS suffix we specify three options, as are listed in descending order of preference:

User input: a user may manually specify the DNS suffix on its own, either to access a 3rd party ALTO service provider or as it does know such information. This input may also origin from a web page where the user downloads the configuration which is loaded as user input.

DHCP: a network provider provides the DNS suffix through a DHCP option.

Reverse DNS: the DNS system can be used to retrieve the DNS suffix through reverse lookup of an FQDN associated with an IP address. This is the last resort if all other options failed.

3. Retrieving the URI by U-NAPTR

This section specifies the U-NAPTR based resolution process. To start the U-NAPTR resolution process a domain name is required as input. Thus the section is devided into two parts: Section 3.1 describes the U-NAPTR resolution process itself. How the client identifies this DNS suffix of the access network where the resource consumer is registered in is described in Section 3.2.

3.1. U-NAPTR Resolution

ALTO servers are identified by U-NAPTR/DDDS (URI-Enabled NAPTR/ Dynamic Delegation Discovery Service) [RFC4848] application unique strings, in the form of a DNS name. An example is 'altoserver.example.com'.

Clients need to use the U-NAPTR [RFC4848] specification described below to obtain a URI (indicating host and protocol) for the applicable ALTO service. In this document, only the HTTP and HTTPS URL schemes are defined, as the ALTO protocol specification defines the access over both protocols, but no other [I-D.ietf-alto-protocol]. Note that the HTTP URL can be any valid HTTP(s) URL, including those containing path elements.

The following two DNS entries show the U-NAPTR resolution for "example.com" to the HTTPS URL https://altoserver.example.com/secure or the HTTP URL http://altoserver.example.com, with the former being preferred.

```
example.com.

IN NAPTR 100  10   "u"    "ALTO:https"
     "!.*!https://altoserver.example.com/secure!"  ""

IN NAPTR 200  10   "u"    "ALTO:http"
     "!.*!http://altoserver.example.com!"  ""
```

3.2. Retrieving the Domain Name

The U-NAPTR resolution process requires a domain name as input. The algorithm that is applied to determine this domain name is described in this section. We specify three different options. In option 1 the user manually configures a specific ALTO service instance that he wants to use. Option 2 defines a DHCP option to allow the network service provider a remote configuration of the client. In option 3 the client tries to get the domain name by performing a reverse DNS lookup on its IP address.

The resource consumer may have private IP addresses and public IP addresses and depending on the deployment it might be necessary to determine for all IP addresses the ALTO server in charge of. To determine its public IP address the resource consumer may need to use STUN[RFC5389] or BEP24[bep24]. For the following examples we assume that the IP address of the resource consumer is a.b.c.d.

3.2.1. Option 1: User input

A user may want to use a third party ALTO service instance. Therefore we allow the user to specify a DNS suffix on its own, for example in a config file option. The DNS suffix given by the user is combined with the IP address of the resource consumer to allow the third party ALTO service to direct the client to a suitable ALTO server based on the location of the client. A possible DNS suffix entered by the user may be:

myaltoprovider.org

This DNS suffix is prepended with the IP address of the resource consumer in reverse order to compose the domain name used for the final U-NAPTR lookup Section 3.1. In case there are multiple ALTO servers deployed, the third party ALTO service instance can direct the ALTO client to the ALTO server closest to the client based on the IP address.

Multiple lookups with different domain names might be necessary to complete the U-NAPTR resolution process. If there is no response for a lookup the domain name is shortened by one part for the succeeding lookup, until a lookup is successful, as for example

d.c.b.a.myaltoprovider.org.

c.b.a.myaltoprovider.org.

b.a.myaltoprovider.org.

a.myaltoprovider.org.

myaltoprovider.org.

3.2.2. Option 2: DHCP

As a second option network operators can configure the domain name to be used for service discovery within an access network. RFC 5986[RFC5986] defines DHCP IPv4 and IPv6 access network domain name options that identify a domain name that is suitable for service discovery within the access network. The ALTO server discovery

procedure uses these DHCP options to retrieve the domain name as an input for the U-NAPTR resolution. One example could be:

example.com

3.2.3. Option 3: Reverse DNS Lookup

The last option to get the domain name is to use a DNS PTR query for the IP address of the resource consumer. The local DNS server resolves the IP address to the FQDN that also contains the DNS suffix for the respective IP address. A possible answer for a PTR lookup for d.c.b.a.in-addr.apra might be, for example:

d-c-b-a.dsl.westcoast.myisp.net

This domain name can be used for the final U-NAPTR lookup Section 3.1. If there is no response to the lookup the domain name is shortened by one part for one succeeding lookup. If there is still no response we consider the reverse lookup being failed. The domain names used for the example as described above are:

d-c-b-a.dsl.westcoast.myisp.net.

dsl.westcoast.myisp.net.

4. Applicability

This section discusses the applicability of the proposed solution with respect to the resource consumer server discovery and the third party deployment scenarios. Each section discusses the proposed steps that are needed to determine the ALTO Server URI.

4.1. Applicability for Resource Consumer Server Discovery

In this scenario the ALTO server discovery procedure is performed by the resource consumer, for example a peer in a P2P system. After the discovery the peer does the ALTO query on its own, or it might share the ALTO server contact information with a third party, for example a tracker, which then executes the ALTO query on behalf of the peer.

To complete the ALTO server discovery process the resource consumer first SHOULD check whether the user has provider the domain name through manual configuration. If this is not the case the next step SHOULD be to check for the access network domain name DHCP option (Section 3.2.2). Finally the client SHOULD try to retrieve the domain name by the last option, the DNS reverse lookup on its IP address as described in Section 3.2.3.

A client can have several candidate IP addresses that it may use for the discovery process. For example if it is located behind a NAT, a private and a public IP address may be used for the discovery process. It depends on the deployment scenario which of the IP addresses is the correct one. Thus it is out-of-scope of this document to specify how exactly the client finds the right IP address. However in the following we list methods that may be used in order to determine these candidate IP addresses. Generally in P2P environments peers already have implemented mechanisms for NAT-traversal. This includes proprietary solutions to determine a peer's public IP address, for example by asking a neighbour peer about its record of the own IP address. Non-proprietary solutions that are favorable include the Session Traversal Utilities for NAT (STUN) [RFC5986] protocol to determine the public address. If the client is behind a residential gateway another option may be to use Universal Plug and Play (UPnP) [UPnP-IGD-WANIPConnection1] or the NAT Port Mapping Protocol (NAT-PMP) [I-D.cheshire-nat-pmp].

In case the ALTO discovery client has determined the domain name through one of the described options it proceedes with the U-NAPTR lookup as described in Section 3.1.

4.2. Applicability for Third Party Server Discovery

In case of the third party server discovery deployment scenario the entity performing the ALTO server discovery process is different from the resource consumer. Typically the resource consumer is a peer whereas the ALTO client is a resource directory which seeks for ALTO guidance on behalf of the peer. Another use case for the third party discovery is an application that looks for ALTO guidance transparently for the resource consumer, for example a CDN.

Here the ALTO server discovery process can also retrieve guidance through the DHCP option or manual user configuration, but only if the provided discovery information is forwarded by the resource consumer to the third party entity. In this case, additional mechanisms for the forwarding of this discovery information need to be specified. However these mechanisms are out of scope of this doument.

If the third party entity cannot obtain this discovery information, the ALTO server discovery process relies on retrieving the domain name used as input to the U-NAPTR lookup through reverse DNS lookup of the IP address of the resource consumer as described in Section 3.2.3. Usually the third party entity already knows the IP address of the resource consumer which was used to establish the initial connection. In general this IP address is a public address, either of the resource consumer or of the last NAT on the path to the ALTO client. This makes the IP address a good candidate for the DNS PTR query. Thus, we expect that the DNS query will be successfully resolved to the FQDN of the domain where the resource consumer is registered in.

In case the resource consumer needs guidance for a different IP address, for example one from a private network, we recommend that the resource consumer discovers the server itself and forwards the ALTO server contact information directly to the third party entity, which in turn can then do the third party ALTO query. Again, forwarding the contact information from the resource consumer to the third party entity is out of scope of this document.

5. Deployment Considerations

The mechanism specified in this document needs some configuration effort in order to work properly.

5.1. Reverse DNS Lookup

Especially the domain name retrieved through the reverse DNS lookup (PTR records) and the U-NAPTR entry need to be coordinated. In this section we discuss this configuration for different scenarios.

5.1.1. Private customers or very small businesses

For private customers and very small businesses that are DSL or cable customers often a dynamically assigned IP address is provisioned. Here, the reverse DNS lookup (PTR records) are controlled by the ISP and they point to the ISP's domain, e.g.:

p5B203EA1.dip.t-dialin.net.

dslb-084-056-144-100.pools.arcor-ip.net.

187-4-222-157.bnut3700.dsl.brasiltelecom.net.br.

65-154-39-69.ispnetbilling.com.

197-151-94-178.pool.ukrtel.net.

In this case, it would be the responsibility of the respective ISP to provide U-NAPTR entries for the DNS suffix without the endhost part, e.g.:

dip.t-dialin.net.

pools.arcor-ip.net.

bnut3700.dsl.brasiltelecom.net.br.

ispnetbilling.com.

pool.ukrtel.net.

5.1.2. Medium-size customer networks

The second class of customers have their own DNS domain but only one single upstream ISP, e.g.:

(1) ISP my-isp.net assigns an IP address a.b.c.d to its customer

(2) The customer decides that reverse mapping for a.b.c.d should be whatever.customerdomain.com

(3) If the customer wants to support ALTO, he has to ask the ISP for the URI of the ISP's ALTO server which can give guidance to a.b.c.d. Assume that ISP replies it is http://altoserver.my-isp.net

(4) The customer establishes a U-NAPTR entry for his domain

```
customerdomain.com.   IN NAPTR 200  10   "u"    "ALTO:http"
     "!.*!http://altoserver.my-isp.net!"  ""
```

5.1.3. Large Customers

For very large customers with multiple upstream connections we assume that they have their very own traffic optimization policies and thus run their own ALTO server anyway. In this case they need to manage their DNS entries accordingly.

5.2. DHCP option for DNS Suffix

Section 3.2.2 describes the usage of a DHCP option which allows the network operator of the network where the ALTO client is attached to, to provide a DNS suffix. However, this assumes that this particular DHCP option is correctly passed from the DHCP server to the actual host with the ALTO client, and that the particular host understands this DHCP option. This memo assumes the client to be able to understand the proposed DHCP option, otherwise there is no further use of the DHCP option, but the client has to use the other proposed mechanisms.

There are well-known issues with the handling of DHCP options in home gateways. One issue is that unkown DHCP options are not passed through some home gateways, effectively eliminating the DHCP option.

Another well-known issues is the usage of home gateway specific DNS suffixes which "override" the DNS suffix provided by the network operator. For instance, a host behind a home gateway may receive a DNS suffix ".local" instead of "example.com". This suffix is not usuable for the server discovery procedure.

[Editor's note: This section needs references about the well-known issues with home gateways and it relates to the FUN activity on home gateways which needs to be explored further.

6. IANA Considerations

This document registers the following U-NAPTR application service tag:

Application Service Tag: ALTO

Defining Publication: The specification contained within this document.

This document registers the following U-NAPTR application protocol tags:

- Application Protocol Tag: http

  Defining Publication: RFC 2616 [RFC2616]

- Application Protocol Tag: https

  Defining Publication: RFC 2818 [RFC2818]

7. Security Considerations

7.1. General

This is still to be done in later revision of this draft, as the draft evolves heavily right now.

7.2. For U-NAPTR

The address of an ALTO server is usually well-known within an access network; therefore, interception of messages does not introduce any specific concerns.

The primary attack against the methods described in this document is one that would lead to impersonation of an ALTO server since a device does not necessarily have a prior relationship with an ALTO server.

An attacker could attempt to compromise ALTO discovery at any of three stages:

1. providing a falsified domain name to be used as input to U-NAPTR

2. altering the DNS records used in U-NAPTR resolution

3. impersonation of the ALTO server

This document focuses on the U-NAPTR resolution process and hence this section discusses the security considerations related to the DNS handling. The security aspects of obtaining the domain name that is used for input to the U-NAPTR process is described in respective documents, such as [RFC5986].

The domain name that is used to authenticated the ALTO server is the domain name in the URI that is the result of the U-NAPTR resolution. Therefore, if an attacker was able to modify or spoof any of the DNS records used in the DDDS resolution, this URI could be replaced by an invalid URI. The application of DNS security (DNSSEC) [RFC4033] provides a means to limit attacks that rely on modification of the DNS records used in U-NAPTR resolution. Security considerations specific to U-NAPTR are described in more detail in [RFC4848].

An "https:" URI is authenticated using the method described in Section 3.1 of [RFC2818]. The domain name used for this authentication is the domain name in the URI resulting from U-NAPTR resolution, not the input domain name as in [RFC3958]. Using the domain name in the URI is more compatible with existing HTTP client software, which authenticate servers based on the domain name in the URI.

An ALTO server that is identified by an "http:" URI cannot be authenticated. If an "http:" URI is the product of the ALTO discovery, this leaves devices vulnerable to several attacks. Lower layer protections, such as layer 2 traffic separation might be used to provide some guarantees.

8. Open Issues

Here are a few open issues to be clarified:

Handling of reverse DNS lookups for IPv6: Refer to [RFC4472] for a discussion about the issues.

Missing reverse DNS entries for an IP address: There may be cases where the reverse DNS lookup does not yield any result. However, this will leave the ALTO client with no choice, other than giving up. This needs better documentation.

How to handled multiple results: For instance, a host behind a NAT that yields an ALTO server in the private IP address domain and one in the public IP address domain. Whom to ask?

Normative Language The current version of this memo lacks the proper normative language in many places.

9. Conclusion

This document describes a general ALTO server discovery process and discusses how the process can be applied in different deployment scenarios, including the resouce consumer discovery as well as the third party discovery.

10. References

10.1. Normative References

[RFC2616] 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.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

10.2. Informative References

[I-D.cheshire-nat-pmp] Cheshire, S., "NAT Port Mapping Protocol (NAT-PMP)", draft-cheshire-nat-pmp-03 (work in progress), April 2008.

[I-D.ietf-alto-deployments] Stiemerling, M. and S. Kiesel, "ALTO Deployment Considerations", draft-ietf-alto-deployments-01 (work in progress), March 2011.

[I-D.ietf-alto-protocol] Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-04 (work in progress), May 2010.

[I-D.ietf-alto-reqs] Kiesel, S., Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", draft-ietf-alto-reqs-08 (work in progress), March 2011.

[RFC4472] Durand, A., Ihren, J., and P. Savola, "Operational Considerations and Issues with IPv6 DNS", RFC 4472, April 2006.

[RFC4848] Daigle, L., "Domain-Based Application Service Location

Using URIs and the Dynamic Delegation Discovery Service (DDDS)", RFC 4848, April 2007.

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

[RFC5986] Thomson, M. and J. Winterbottom, "Discovering the Local Location Information Server (LIS)", RFC 5986, September 2010.

[UPnP-IGD-WANIPConnection1] UPnP Forum, "Internet Gateway Device (IGD) Standardized Device Control Protocol V 1.0: WANIPConnection:1 Service Template Version 1.01 For UPnP Version 1.0", DCP 05-001, November 2001.

[bep24] Harrison, D., "Tracker Returns External IP", BEP http://bittorrent.org/beps/bep_0024.html.

Appendix A. Contributors List and Acknowledgments

The initial version of this document was co-authored by Marco Tomsu <marco.tomsu@alcatel-lucent.com>.

Hannes Tschofenig provided the initial input to the U-NAPTR solution part. Hannes and Martin Thomson provided excellent feedback and input to the server discovery.

The authors would also like to thank the following persons for their contribution to this document or its predecessors: Richard Alimi, David Bryan, Roni Even, Gustavo Garcia, Jay Gu, Xingfeng Jiang, Enrico Marocco, Victor Pascual, Y. Richard Yang, Yu-Shun Wang, Yunfei Zhang, Ning Zong.

Marco Tomsu and Nico Schwan are partially supported by the ENVISION project (http://www.envision-project.org), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ENVISION project or the European Commission.

Michael Scharf is supported by the German-Lab project (http://www.german-lab.de) funded by the German Federal Ministry of Education and Research (BMBF).

Martin Stiemerling is partially supported by the COAST project (COntent Aware Searching, retrieval and sTreaming, http://www.coast-fp7.eu), a research project supported by the European Commission under its 7th Framework Program (contract no. 248036). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the COAST project or the European Commission.

Authors' Addresses

Sebastian Kiesel
University of Stuttgart Computing Center
Allmandring 30
Stuttgart 70550
Germany

Email: ietf-alto@skiesel.de
URI: http://www.rus.uni-stuttgart.de/nks/

Martin Stiemerling
NEC Laboratories Europe
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 4342 113
Email: martin.stiemerling@neclab.eu
URI: http://ietf.stiemerling.org

Nico Schwan
Alcatel-Lucent Bell Labs
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: nico.schwan@alcatel-lucent.com
URI: www.alcatel-lucent.com/bell-labs

Michael Scharf
Alcatel-Lucent Bell Labs
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: michael.scharf@alcatel-lucent.com
URI: www.alcatel-lucent.com/bell-labs

Haibin Song
Huawei

Email: melodysong@huawei.com

ALTO
Internet-Draft
Intended status: Standards Track
Expires: March 17, 2012

S. Kiesel
University of Stuttgart
M. Stiemerling
NEC Europe Ltd.
N. Schwan
M. Scharf
Alcatel-Lucent Bell Labs
H. Song
Huawei

September 14, 2011

ALTO Server Discovery
draft-ietf-alto-server-discovery-02

Abstract

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource.

Entities seeking guidance need to discover and possibly select an ALTO server to ask. This is called ALTO server discovery. This memo describes an ALTO server discovery mechanism based on several alternative mechanisms that are applicable in a diverse set of ALTO deployment scenarios.

Status of this Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on March 17, 2012.

Copyright Notice

Table of Contents

1. Introduction

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications, which have to select one or several hosts from a set of candidates, that are able to provide a desired resource [RFC5693]. The requirements for ALTO are itemized in [I-D.ietf-alto-reqs].

ALTO is realized by a client-server protocol. ALTO clients send queries to ALTO servers, in order to solicit guidance. Hence, ALTO clients need to know the contact information of ALTO servers, which can provide appropriate guidance for a given resource consumer. This information can be retrieved by invoking the ALTO discovery procedure defined in this document.

The ALTO protocol specification [I-D.ietf-alto-protocol] is based on HTTP. Therefore, it expects that the ALTO discovery procedure yields the HTTP(S) URI of the ALTO server's Information Resource Directory, which gives further information about the capabilities and services provided by that ALTO server. Further (DNS) lookups may be necessary in order to find out the ALTO server's IP address.

There are various architectural options where to place the ALTO client and the ALTO server discovery procedure:

- One option is that the ALTO client and the ALTO server discovery procedure are embedded directly in the resource consumer, i.e., the application protocol entity that will eventually initiate data transmission to/from the selected resource provider(s). In this case, the ALTO server discovery procedure might be able to interact with the user (i.e., prompt for a host name). Furthermore, it may use services such as DHCP, which are only available within the access network to which the resource consumer is connected.

- Another option is to integrate the ALTO client and the ALTO server discovery procedure into a third party such as a resource directory ("peer-to-peer tracker"), which issues ALTO queries on behalf of various resource consumers. This third party may reside in a different part of the network (administrative domain) than the resource consumer. It may occur that said third party whishes to issue ALTO queries on behalf of a resouce consumer, but all it knows about the resource consumer is the source IP address of messages originating from it (i.e., the resource consumer's IP address or the "public" IP address of the outermost NAT in front of the resource consumer). This IP address will be the only input parameter to the ALTO server discovery procedure, which will have to find an ALTO server that can give appropriate guidance for that

resource consumer.

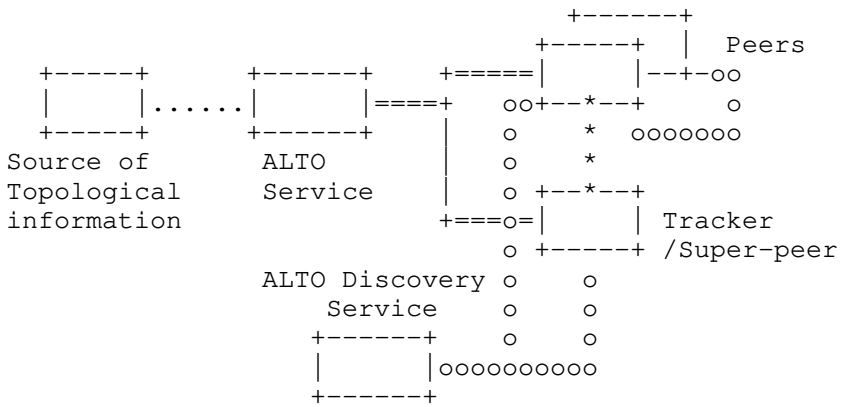
A more detailed discussion of various options where to place the funcional entities comprising the overall ALTO architecture can be found in [I-D.ietf-alto-deployments].

The goal of this memo is to propose a uniform mechanism for all types of ALTO client deployments that is implementable and deployable at a fast pace, i.e., without creating other deployment dependencies for ALTO. We propose a schema which employs the UNAPTR mechanism [RFC4848] to determine the URI of the ALTO server and where mutliple input methods to the UNAPTR process can be used.

Comments and discussions about this memo should be directed to the ALTO working group: alto@ietf.org.

1.1. Discovery Scenarios

Figure 1 below shows an overview on the different entities of a generic ALTO framework. The ALTO Server discovery mechanism is used by the peer-to-peer (P2P) application in order retrieve the point of contact of the ALTO Service.



Legend:

===	ALTO query protocol
ooo	ALTO service discovery protocol
***	Application protocol (out of scope)
...	Provisioning or initialization (out of scope)

Figure 1: ALTO Discovery Overview

Hereby the ALTO service discovery scenarios are classified into two types: one is the ALTO server discovery by the resource consumer, and the other is the ALTO server discovery by a third party, such as application trackers. Before the specification of the discovery mechanism the following section illustrates and discusses both scenarios.

1.1.1. ALTO Server Discovery by Resource Consumers

The ALTO service discovery in some scenarios needs to be performed by the resource consumer itself. In particular in P2P applications without a tracker like DHTs and other conventional client/server applications.

In addition also P2P application which are tracker based may embed the ALTO client into the resource consumer to allow peers a selection of peers after retrieving the peer list from the application tracker. Another option is that the resource consumer peer sends its ALTO server address information to the application tracker or any other third party entity, which in turn will contact the specific ALTO server in order to retrieve ALTO guidance on behalf of the resource consumer.

The following figure illustrates this scenario, showing the relationship between the different entities as discussed before.

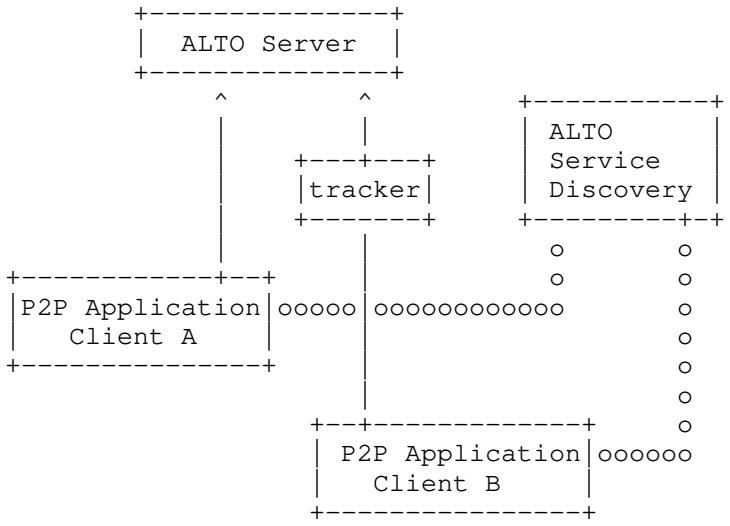


Figure 2: Resource Consumer ALTO Server Discovery (Example)

1.1.2. ALTO Server Discovery by a Third Party

Some P2P applications have trackers, and these applications might not need to have their clients looking for the ALTO server guidance. In these scenarios trackers query the ALTO servers for guidance themselves, and then return the final ranked result to the application clients. However, application clients are distributed among different network operators and autonomous systems. Trackers thus need to find different ALTO servers for the clients located in different operator networks or autonomous systems. In such scenarios the discovery is thus not performed by the resource consumer, but a third party entity on behalf of the resource consumer.

Figure 3 shows an example for a third party ALTO server discovery. For Client1 (1), the tracker has not cached yet the mapping between Client1's network operator and its ALTO server address, so it uses the ALTO Discovery Service to determine the address of the ALTO server in that operator's domain (2). Then the tracker interacts with ALTO Server1 (3)(4) on behalf of Client1 (to get the network map and cost map), finally, the ranked list is sent back to Client1 (5). For Client2, the tracker has cached the mapping between Client2's network operator and ALTO Server2's address, so it does not need to perform the discovery process (which are the labels (a),(b), (c), and (d)). If the application tracker already has the network map and cost map from ALTO Server2, then it does not need to query the ALTO Server for network map and cost map frequently.

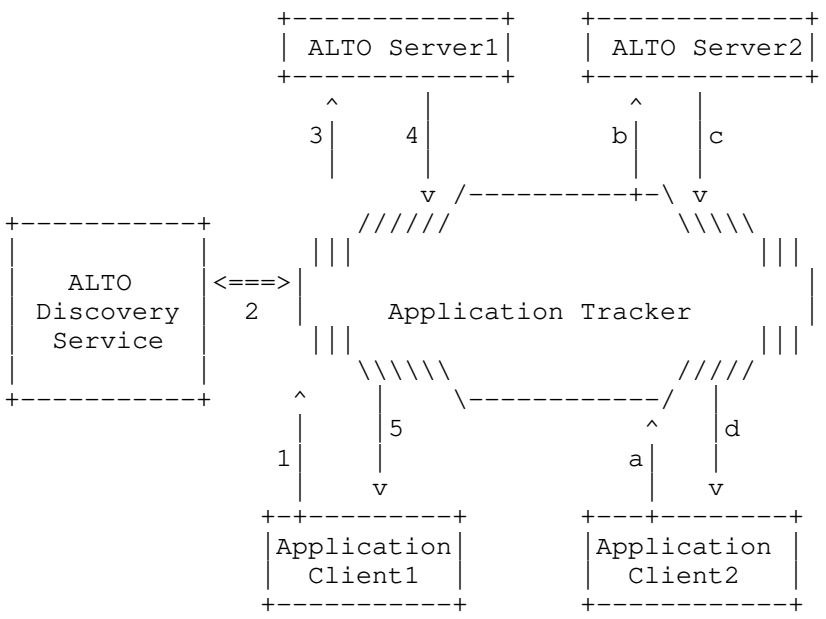


Figure 3: Third Party ALTO Server Discovery (Example)

1.2. Pre-Conditions

The whole document assumes certain pre-conditions, in particular:

- The ALTO server discovery procedure is executed on a per IP address base. Multiple IP addresses per interface or multiple IP addresses assigned to different IP interfaces require to repeat the procedure for every IP address. It may be fine to group IP addresses according their domain suffixes and to perfom the procedure for such a group. However, this is out of scope of this document.[Editor's note: this may relate to the work of the MIF WG]

- The ALTO server discovery procedure is executed on a per IP family base, i.e., seperate for IPv4 and IPv6. It is up to the ALTO client to decide which of the possible multiple results of different IP address families to use. The choice of whether to use IPv4 or IPv6 is out of scope of this document.

- A change of the IP address at an interface invalidates the result of the ALTO server discovery procedure. For instance, if the IP address assigned to a mobile host changes due to host mobility, it is required to run the ALTO server discovery procedure for the new IP address without relying on earlier gained information.

2. Protocol Overview

We define multiple alternatives to discover the IP address of the ALTO server, as there are a number of ways possible how such information can be provided to the ALTO client. The choice of method is up to the local network deployment. For instance, there can be deployments where the ALTO server in charge for ALTO client is provisioned by the network operator and communicated to the ALTO client's host via a DHCP option, while in other deployments no such means may exist. It should be noted that there is no silver bullet solution to the ALTO server discovery, as there too many deployment scenarios in the server discovery space.

The following figure illustrates the different protocols that are used to find the URI of a suitable ALTO server.

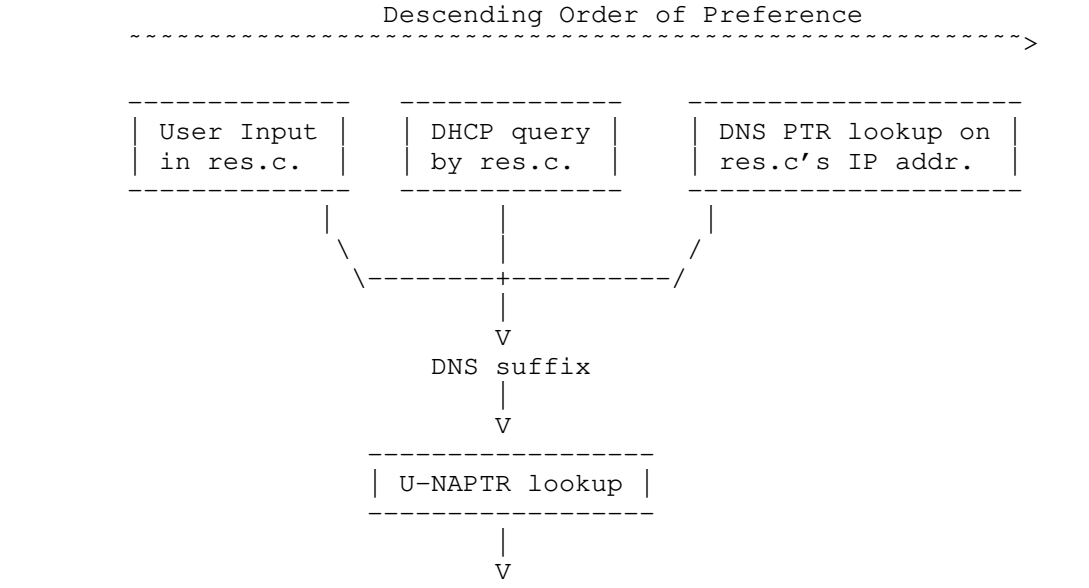


Figure 4: Protocol Overview

Figure 4 illustrates the U-NAPTR based resolution process to retrieve the ALTO Server URL. As a precondition for resolution the U-NAPTR process needs the right domain name as input. This domain name is determined by the IP address of the client and the DNS suffix of the access network where the client is registered in. In order to retrieve the DNS suffix we specify three options, as are listed in descending order of preference:

User input: a user may manually specify the DNS suffix on its own, either to access a 3rd party ALTO service provider or as it does know such information. This input may also origin from a web page where the user downloads the configuration which is loaded as user input.

DHCP: a network provider provides the DNS suffix through a DHCP option.

Reverse DNS: the DNS system can be used to retrieve the DNS suffix through reverse lookup of an FQDN associated with an IP address. This is the last resort if all other options failed.

3. Retrieving the URI by U-NAPTR

This section specifies the U-NAPTR based resolution process. To start the U-NAPTR resolution process a domain name is required as input. Thus the section is devided into two parts: Section 3.2 describes the U-NAPTR resolution process itself. How the client identifies this DNS suffix of the access network where the resource consumer is registered in is described in Section 3.1.

3.1. Retrieving the Domain Name

The U-NAPTR resolution process requires a domain name as input. The algorithm that is applied to determine this domain name is described in this section. We specify three different options. In option 1 the user manually configures a specific ALTO service instance that he wants to use. Option 2 defines a DHCP option to allow the network service provider a remote configuration of the client. In option 3 the client tries to get the domain name by performing a reverse DNS lookup on its IP address.

The resource consumer may have private IP addresses and public IP addresses and depending on the deployment it might be necessary to determine for all IP addresses the ALTO server in charge of. To determine its public IP address the resource consumer may need to use STUN[RFC5389] or BEP24[bep24]. For the following examples we assume that the IP address of the resource consumer is a.b.c.d.

3.1.1. Option 1: User input

A user may want to use a third party ALTO service instance. Therefore we allow the user to specify a DNS suffix on its own, for example in a config file option. The DNS suffix given by the user is combined with the IP address of the resource consumer to allow the third party ALTO service to direct the client to a suitable ALTO server based on the location of the client. A possible DNS suffix entered by the user may be:

```
myaltoprovider.org
```

This DNS suffix is prepended with the IP address of the resource consumer in reverse order to compose the domain name used for the final U-NAPTR lookup Section 3.2. In case there are multiple ALTO servers deployed, the third party ALTO service instance can direct the ALTO client to the ALTO server closest to the client based on the IP address.

Multiple lookups with different domain names might be necessary to complete the U-NAPTR resolution process. If there is no response for

a lookup the domain name is shortened by one part for the succeeding lookup, until a lookup is successful, as for example

d.c.b.a.myaltoprovider.org.

c.b.a.myaltoprovider.org.

b.a.myaltoprovider.org.

a.myaltoprovider.org.

myaltoprovider.org.

3.1.2. Option 2: DHCP

As a second option network operators can configure the domain name to be used for service discovery within an access network. RFC 5986[RFC5986] defines DHCP IPv4 and IPv6 access network domain name options that identify a domain name that is suitable for service discovery within the access network. The ALTO server discovery procedure uses these DHCP options to retrieve the domain name as an input for the U-NAPTR resolution. One example could be:

example.com

3.1.3. Option 3: Reverse DNS Lookup

The last option to get the domain name is to use a DNS PTR query for the IP address of the resource consumer. The local DNS server resolves the IP address to the FQDN that also contains the DNS suffix for the respective IP address. A possible answer for a PTR lookup for d.c.b.a.in-addr.apra might be, for example:

d-c-b-a.dsl.westcoast.myisp.net

This domain name can be used for the final U-NAPTR lookup Section 3.2. If there is no response to the lookup the domain name is shortened by one part for one succeeding lookup. If there is still no response we consider the reverse lookup being failed. The domain names used for the example as described above are:

d-c-b-a.dsl.westcoast.myisp.net.

dsl.westcoast.myisp.net.

3.2. U-NAPTR Resolution

The ALTO protocol specification [I-D.ietf-alto-protocol] , it expects that the ALTO discovery procedure yields the HTTP(S) URI of the ALTO server's Information Resource Directory, which gives further information about the capabilities and services provided by that ALTO server. The first step of the ALTO server discovery procedure (see Section 3.1) yielded an U-NAPTR/DDDS (URI-Enabled NAPTR/Dynamic Delegation Discovery Service) [RFC4848] application unique strings, in the form of a DNS name. An example is "example.com".

In the second step, the ALTO Server discovery procedure needs to use the U-NAPTR [RFC4848] specification described below to obtain a URI (indicating host and protocol) for the ALTO server's Information Resource Directory. In this document, only the HTTP and HTTPS URL schemes are defined, as the ALTO protocol specification defines the access over both protocols, but no other [I-D.ietf-alto-protocol]. Note that the HTTP URL can be any valid HTTP(s) URL, including those containing path elements.

The following two DNS entries show the U-NAPTR resolution for "example.com" to the HTTPS URL https://altoserver.example.com/secure/directory or the HTTP URL http://altoserver.example.com/directory, with the former being preferred.

```
example.com.

IN NAPTR 100  10   "u"    "ALTO:https"
     "!.*!https://altoserver.example.com/secure/directory!"  ""

IN NAPTR 200  10   "u"    "ALTO:http"
     "!.*!http://altoserver.example.com/directory!"  ""
```

4. Applicability

This section discusses the applicability of the proposed solution with respect to the resource consumer server discovery and the third party deployment scenarios. Each section discusses the proposed steps that are needed to determine the ALTO Server URI.

4.1. Applicability for Resource Consumer Server Discovery

In this scenario the ALTO server discovery procedure is performed by the resource consumer, for example a peer in a P2P system. After the discovery the peer does the ALTO query on its own, or it might share the ALTO server contact information with a third party, for example a tracker, which then executes the ALTO query on behalf of the peer.

To complete the ALTO server discovery process the resource consumer first SHOULD check whether the user has provider the domain name through manual configuration. If this is not the case the next step SHOULD be to check for the access network domain name DHCP option (Section 3.1.2). Finally the client SHOULD try to retrieve the domain name by the last option, the DNS reverse lookup on its IP address as described in Section 3.1.3.

A client can have several candidate IP addresses that it may use for the discovery process. For example if it is located behind a NAT, a private and a public IP address may be used for the discovery process. It depends on the deployment scenario which of the IP addresses is the correct one. Thus it is out-of-scope of this document to specify how exactly the client finds the right IP address. However in the following we list methods that may be used in order to determine these candidate IP addresses. Generally in P2P environments peers already have implemented mechanisms for NAT-traversal. This includes proprietary solutions to determine a peer's public IP address, for example by asking a neighbour peer about its record of the own IP address. Non-proprietary solutions that are favorable include the Session Traversal Utilities for NAT (STUN) [RFC5986] protocol to determine the public address. If the client is behind a residential gateway another option may be to use Universal Plug and Play (UPnP) [UPnP-IGD-WANIPConnection1] or the NAT Port Mapping Protocol (NAT-PMP) [I-D.cheshire-nat-pmp].

In case the ALTO discovery client has determined the domain name through one of the described options it proceedes with the U-NAPTR lookup as described in Section 3.2.

4.2. Applicability for Third Party Server Discovery

In case of the third party server discovery deployment scenario the entity performing the ALTO server discovery process is different from the resource consumer. Typically the resource consumer is a peer whereas the ALTO client is a resource directory which seeks for ALTO guidance on behalf of the peer. Another use case for the third party discovery is an application that looks for ALTO guidance transparently for the resource consumer, for example a CDN.

Here the ALTO server discovery process can also retrieve guidance through the DHCP option or manual user configuration, but only if the provided discovery information is forwarded by the resource consumer to the third party entity. In this case, additional mechanisms for the forwarding of this discovery information need to be specified. However these mechanisms are out of scope of this doument.

If the third party entity cannot obtain this discovery information, the ALTO server discovery process relies on retrieving the domain name used as input to the U-NAPTR lookup through reverse DNS lookup of the IP address of the resource consumer as described in Section 3.1.3. Usually the third party entity already knows the IP address of the resource consumer which was used to establish the initial connection. In general this IP address is a public address, either of the resource consumer or of the last NAT on the path to the ALTO client. This makes the IP address a good candidate for the DNS PTR query. Thus, we expect that the DNS query will be successfully resolved to the FQDN of the domain where the resource consumer is registered in.

In case the resource consumer needs guidance for a different IP address, for example one from a private network, we recommend that the resource consumer discovers the server itself and forwards the ALTO server contact information directly to the third party entity, which in turn can then do the third party ALTO query. Again, forwarding the contact information from the resource consumer to the third party entity is out of scope of this document.

5. Deployment Considerations

The mechanism specified in this document needs some configuration effort in order to work properly.

5.1. Reverse DNS Lookup

Especially the domain name retrieved through the reverse DNS lookup (PTR records) and the U-NAPTR entry need to be coordinated. In this section we discuss this configuration for different scenarios.

5.1.1. Private customers or very small businesses

For private customers and very small businesses that are DSL or cable customers often a dynamically assigned IP address is provisioned. Here, the reverse DNS lookup (PTR records) are controlled by the ISP and they point to the ISP's domain, e.g.:

p5B203EA1.dip.t-dialin.net.

dslb-084-056-144-100.pools.arcor-ip.net.

187-4-222-157.bnut3700.dsl.brasiltelecom.net.br.

65-154-39-69.ispnetbilling.com.

197-151-94-178.pool.ukrtel.net.

In this case, it would be the responsibility of the respective ISP to provide U-NAPTR entries for the DNS suffix without the endhost part, e.g.:

dip.t-dialin.net.

pools.arcor-ip.net.

bnut3700.dsl.brasiltelecom.net.br.

ispnetbilling.com.

pool.ukrtel.net.

5.1.2. Medium-size customer networks

The second class of customers have their own DNS domain but only one single upstream ISP, e.g.:

(1) ISP my-isp.net assigns an IP address a.b.c.d to its customer

(2) The customer decides that reverse mapping for a.b.c.d should be whatever.customerdomain.com

(3) If the customer wants to support ALTO, he has to ask the ISP for the URI of the ISP's ALTO server which can give guidance to a.b.c.d. Assume that ISP replies it is http://altoserver.my-isp.net

(4) The customer establishes a U-NAPTR entry for his domain

```
customerdomain.com.   IN NAPTR 200  10   "u"    "ALTO:http"
     "!.*!http://altoserver.my-isp.net!"  ""
```

5.1.3. Large Customers

For very large customers with multiple upstream connections we assume that they have their very own traffic optimization policies and thus run their own ALTO server anyway. In this case they need to manage their DNS entries accordingly.

5.2. DHCP option for DNS Suffix

Section 3.1.2 describes the usage of a DHCP option which allows the network operator of the network where the ALTO client is attached to, to provide a DNS suffix. However, this assumes that this particular DHCP option is correctly passed from the DHCP server to the actual host with the ALTO client, and that the particular host understands this DHCP option. This memo assumes the client to be able to understand the proposed DHCP option, otherwise there is no further use of the DHCP option, but the client has to use the other proposed mechanisms.

There are well-known issues with the handling of DHCP options in home gateways. One issue is that unkown DHCP options are not passed through some home gateways, effectively eliminating the DHCP option.

Another well-known issues is the usage of home gateway specific DNS suffixes which "override" the DNS suffix provided by the network operator. For instance, a host behind a home gateway may receive a DNS suffix ".local" instead of "example.com". This suffix is not usuable for the server discovery procedure.

[Editor's note: This section needs references about the well-known issues with home gateways and it relates to the FUN activity on home gateways which needs to be explored further.

6. IANA Considerations

This document registers the following U-NAPTR application service tag:

Application Service Tag: ALTO

Defining Publication: The specification contained within this document.

This document registers the following U-NAPTR application protocol tags:

- Application Protocol Tag: http

  Defining Publication: RFC 2616 [RFC2616]

- Application Protocol Tag: https

  Defining Publication: RFC 2818 [RFC2818]

7. Security Considerations

7.1. General

This is still to be done in later revision of this draft, as the draft evolves heavily right now.

7.2. For U-NAPTR

The address of an ALTO server is usually well-known within an access network; therefore, interception of messages does not introduce any specific concerns.

The primary attack against the methods described in this document is one that would lead to impersonation of an ALTO server since a device does not necessarily have a prior relationship with an ALTO server.

An attacker could attempt to compromise ALTO discovery at any of three stages:

1. providing a falsified domain name to be used as input to U-NAPTR

2. altering the DNS records used in U-NAPTR resolution

3. impersonation of the ALTO server

This document focuses on the U-NAPTR resolution process and hence this section discusses the security considerations related to the DNS handling. The security aspects of obtaining the domain name that is used for input to the U-NAPTR process is described in respective documents, such as [RFC5986].

The domain name that is used to authenticated the ALTO server is the domain name in the URI that is the result of the U-NAPTR resolution. Therefore, if an attacker was able to modify or spoof any of the DNS records used in the DDDS resolution, this URI could be replaced by an invalid URI. The application of DNS security (DNSSEC) [RFC4033] provides a means to limit attacks that rely on modification of the DNS records used in U-NAPTR resolution. Security considerations specific to U-NAPTR are described in more detail in [RFC4848].

An "https:" URI is authenticated using the method described in Section 3.1 of [RFC2818]. The domain name used for this authentication is the domain name in the URI resulting from U-NAPTR resolution, not the input domain name as in [RFC3958]. Using the domain name in the URI is more compatible with existing HTTP client software, which authenticate servers based on the domain name in the URI.

An ALTO server that is identified by an "http:" URI cannot be authenticated. If an "http:" URI is the product of the ALTO discovery, this leaves devices vulnerable to several attacks. Lower layer protections, such as layer 2 traffic separation might be used to provide some guarantees.

8. Open Issues

Here are a few open issues to be clarified:

Handling of reverse DNS lookups for IPv6: Refer to [RFC4472] for a discussion about the issues.

Missing reverse DNS entries for an IP address: There may be cases where the reverse DNS lookup does not yield any result. However, this will leave the ALTO client with no choice, other than giving up. This needs better documentation.

How to handled multiple results: For instance, a host behind a NAT that yields an ALTO server in the private IP address domain and one in the public IP address domain. Whom to ask?

Normative Language The current version of this memo lacks the proper normative language in many places.

9. Conclusion

This document describes a general ALTO server discovery process and discusses how the process can be applied in different deployment scenarios, including the resouce consumer discovery as well as the third party discovery.

10. References

10.1. Normative References

[RFC2616] 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.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

10.2. Informative References

[I-D.cheshire-nat-pmp] Cheshire, S., "NAT Port Mapping Protocol (NAT-PMP)", draft-cheshire-nat-pmp-03 (work in progress), April 2008.

[I-D.ietf-alto-deployments] Stiemerling, M. and S. Kiesel, "ALTO Deployment Considerations", draft-ietf-alto-deployments-02 (work in progress), July 2011.

[I-D.ietf-alto-protocol] Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-09 (work in progress), June 2011.

[I-D.ietf-alto-reqs] Kiesel, S., Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", draft-ietf-alto-reqs-11 (work in progress), July 2011.

[RFC4472] Durand, A., Ihren, J., and P. Savola, "Operational Considerations and Issues with IPv6 DNS", RFC 4472, April 2006.

[RFC4848] Daigle, L., "Domain-Based Application Service Location

Using URIs and the Dynamic Delegation Discovery Service (DDDS)", RFC 4848, April 2007.

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

[RFC5986] Thomson, M. and J. Winterbottom, "Discovering the Local Location Information Server (LIS)", RFC 5986, September 2010.

[UPnP-IGD-WANIPConnection1] UPnP Forum, "Internet Gateway Device (IGD) Standardized Device Control Protocol V 1.0: WANIPConnection:1 Service Template Version 1.01 For UPnP Version 1.0", DCP 05-001, November 2001.

[bep24] Harrison, D., "Tracker Returns External IP", BEP http://bittorrent.org/beps/bep_0024.html.

Appendix A. Contributors List and Acknowledgments

The initial version of this document was co-authored by Marco Tomsu <marco.tomsu@alcatel-lucent.com>.

Hannes Tschofenig provided the initial input to the U-NAPTR solution part. Hannes and Martin Thomson provided excellent feedback and input to the server discovery.

The authors would also like to thank the following persons for their contribution to this document or its predecessors: Richard Alimi, David Bryan, Roni Even, Gustavo Garcia, Jay Gu, Xingfeng Jiang, Enrico Marocco, Victor Pascual, Y. Richard Yang, Yu-Shun Wang, Yunfei Zhang, Ning Zong.

Marco Tomsu and Nico Schwan are partially supported by the ENVISION project (http://www.envision-project.org), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ENVISION project or the European Commission.

Michael Scharf is supported by the German-Lab project (http://www.german-lab.de) funded by the German Federal Ministry of Education and Research (BMBF).

Martin Stiemerling is partially supported by the COAST project (COntent Aware Searching, retrieval and sTreaming, http://www.coast-fp7.eu), a research project supported by the European Commission under its 7th Framework Program (contract no. 248036). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the COAST project or the European Commission.

Authors' Addresses

Sebastian Kiesel
University of Stuttgart Computing Center
Allmandring 30
Stuttgart 70550
Germany

Email: ietf-alto@skiesel.de
URI: http://www.rus.uni-stuttgart.de/nks/

Martin Stiemerling
NEC Laboratories Europe
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 4342 113
Email: martin.stiemerling@neclab.eu
URI: http://ietf.stiemerling.org

Nico Schwan
Alcatel-Lucent Bell Labs
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: nico.schwan@alcatel-lucent.com
URI: www.alcatel-lucent.com/bell-labs

Michael Scharf
Alcatel-Lucent Bell Labs
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: michael.scharf@alcatel-lucent.com
URI: www.alcatel-lucent.com/bell-labs

Haibin Song
Huawei

Email: melodysong@huawei.com

ALTO
Internet-Draft
Intended Status: Informational
Expires: April 23, 2012

Fabio Picconi
Technicolor

October 21, 2011

ALTO home proxy
draft-picconi-alto-home-proxy-00

Abstract

This documents discusses the use of ALTO proxies running on home devices such as gateways or home NAS servers. An ALTO home proxy caches ALTO information obtained from an origin ALTO server, and uses that information to serve queries originating from the home network. ALTO home proxies reduce ALTO traffic and query latency, improve scalability, and enhance user privacy.

Table of Contents

1 Introduction

The ALTO service allows applications to obtain information about the network cost between two peers. Applications use this information to optimize their traffic (e.g., through proximity-based neighbor selection), leading to better performance for end users, and potentially lower operational costs for network operators. See [RFC5693] for the ALTO problem statement.

The ALTO protocol [I-D.ietf-alto-protocol] defines communications between an ALTO client (requesting cost information) and an ALTO server (providing such information). Clients are typically peer-to-peer applications run by users, while servers are typically run by network operators. Peer-to-peer trackers may also behave as ALTO clients, using the retrieved ALTO information to return localized peer sets. This document considers only the first scenario, in which peers contact an ALTO server directly to obtain ALTO information.

To improve scalability, peers may exchange the cost information retrieved from an ALTO server, thus reducing server load. This is known as ALTO information redistribution, and is discussed in [I-D.ietf-alto-protocol] and [I-D.gu-alto-redistribution]. The ALTO protocol does not make redistribution mandatory, and does not define how it should be performed. It is up to each application to decide whether to implement redistribution, and how (e.g., by using DHT or a gossip-based protocol).

In this document we argue that distribution of ALTO information can be enhanced by employing home proxies. An ALTO home proxy caches ALTO information obtained from an origin ALTO server, and uses it to respond to ALTO queries originating from the local home network. This presents several advantages compared to using a remotely located ALTO server:

1) Reduced ALTO traffic. Application-dependent redistribution might result in unnecessary ALTO traffic. Consider, for instance, a home network with two different active applications (e.g., a file sharing application and a streaming application) which both implement redistribution but using different mechanisms. The network and cost maps downloaded by one application cannot be shared with the other. If both applications use the ALTO home proxy, the ALTO information will be fetched only once.

2) Increased redistribution. ALTO home proxies may redistribute information to the home proxies of other peers. Therefore, applications which do not implement redistribution may still generate little or no load on the ALTO server if they use an ALTO home proxy that implements redistribution.

3) Reduced latency. In case of a cache hit, the ALTO home proxy can serve queries quickly given the low RTT and high bandwidth of home networks. This may improve the performance of delay-sensitive applications.

4) Background updates. If the home proxy is executed in a device that is always on and connected to the network (such as a home gateway or a home NAS), ALTO information updates can be pushed to the proxy during off-peak hours, when there is plenty of available bandwidth and ALTO servers are lightly loaded.

5) Discovery. If the home proxy has been configured to use a default origin ALTO server, then applications using the proxy do not need to use ALTO discovery to select an ALTO server. Moreover, applications may easily discover the home proxy using protocols such as SLP [RFC2608], uPnP, or Bonjour.

6) User privacy. Serving ALTO queries within the home makes it more difficult for a third party (e.g., a remote ALTO server) to infer the user's communications patterns by analyzing ALTO traffic.

The following sections discuss in more detail several aspects of using ALTO home proxies.

2. Devices to run home proxies

ALTO home proxies may run on any home device which is connected to the home network and has Internet access. This includes PCs, home NAS servers, home gateways, etc. However, it may be preferable to run home proxies on devices which are always on, such as a home gateway or a NAS. This allows the proxy to optimize traffic (e.g., download ALTO information updates during off-peak hours), and to maximize the number of queries that can be served locally.

Many users employ home gateways which are owned and managed by their Internet Service Provider (ISP). Typically, the ISP ships the gateway to the user when he or she signs up for the Internet access plan. The gateway is remotely configured by the ISP, which retains control on the firmware, and in particular all the services, that run on the gateway. This is typically the case for triple-play subscriptions, where the gateway runs VoIP and IPTV services in addition to Internet access.

ISPs that deploy ALTO servers in their network will probably consider implementing ALTO home proxies as well. The computing power of current gateways should easily support the small increase in CPU load caused by the ALTO proxy service, especially given that the service only replies to queries originating from the local network. Moreover, the home proxy can be added to already deployed gateways by means of a remote firmware upgrade.

Some users employ gateways which are not managed by their ISP. In this case, gateway vendors may provide firmware upgrades which can be installed by the user. Similarly, vendors or home NAS servers or other networked home devices may provide firmware upgrades which include an ALTO home proxy.

3. Proxy configuration and discovery

Two steps must be executed before applications can use an ALTO home proxy: configuration and discovery.

3.1. Configuration

This step involves configuring the ALTO home proxy parameters. One important parameter is the default origin ALTO server, i.e., the server from which ALTO information will be fetched and cached, and which will be used by default to answer queries. This can be obtained using the mechanisms described in [I-D.ietf-alto-server-discovery]. If the ALTO home proxy runs on an ISP-managed gateway, the ISP may also provide a default origin ALTO proxy (the ISP's own).

Another important parameter is whether the proxy will behave as a caching proxy or a local server (see Section 3.1 for more details).

Other parameters may specify the maximum amount of data that can be cached, the expiration time, etc. The user may be able to configure these parameters through a local management interface (e.g., a web page).

3.2. Proxy discovery

Applications running on the home can discover a local ALTO home proxy by employing protocols such as SLP [RFC2608], uPnP, or Bonjour. An important point is that application need not apply the discovery mechanisms that are needed to use a remote ALTO server [I-D.ietf-alto-server-discovery], as the home proxy has already been configured with a default origin ALTO server.

4. Proxy behavior

Similarly to HTTP proxies, ALTO home proxies may be transparent or non-transparent. Also, ALTO home proxies may behave as caching proxies or local ALTO servers.

4.1. Transparent vs. non-transparent

A non-transparent proxy requires ALTO clients to be aware of their presence, and direct ALTO queries to them instead of remote ALTO servers. ALTO clients must be configured, either manually or automatically, to use the home proxy. Manual configuration is typically achieved by prompting the user for an IP address belonging to the home network (e.g., 192.168.0.x). Automatic configuration can be achieved by using a discovery protocol such as SLP, uPnP, or Bonjour.

A transparent proxy intercepts ALTO requests directed to a remote ALTO server. Therefore, they do not require ALTO clients to be aware of their presence, or to discover them using protocols such as SLP,

uPnP, or Bonjour. Transparent proxies would typically run in home gateways so that they can intercept requests originating from multiple devices of the home network.

4.2. Caching proxies vs. local ALTO servers

ALTO home proxies may behave as simple HTTP caching proxies. The main advantage of this is caching large responses from ALTO servers, such as full network and cost maps. A caching proxy may also use redistribution to fetch cacheable responses from other caching proxies, thus reducing the load on the ALTO server. Caching proxies may also prefetch ALTO information (e.g., an updated version of the cost map) to reduce latency.

The disadvantage of caching proxies is that client-specific requests, such as filtered maps, will probably result in a cache miss. To counter this limitation, a home proxy may behave as a local ALTO server. In this case, it uses the entire cost and network maps previously downloaded from the remote ALTO server to produce, whenever possible, a valid response to the client. A proxy acting as a local server may return filtered maps even though such response has not been previously cached.

The main limitation of proxies acting as local ALTO server is that not all the functionalities of the origin ALTO server may be available, e.g., generating signed data, or obtaining information based on non-standard cost models which are not publicly known.

5. Obtaining ALTO information

Besides from caching responses from ALTO servers, home proxies may obtain ALTO information from wide variety of sources. Update dissemination may be pull-based or push-based [I-D.gu-alto-redistribution], and peer-to-peer protocols such as BitTorrent may be used for redistribution between home proxies.

An important issue is deciding which information to obtain, i.e., which resources and from which ALTO servers. Some users may manually configure their applications to use an ALTO server different from that obtained through the standard ALTO discovery mechanism. Home proxies should be aware of such scenarios, and obtain the appropriate ALTO information to maximize their hit ratio.

6 Security Considerations

ALTO home proxying loses its benefits when clients employ SSL/TLS to

verify the identity of the remote ALTO server. Clients may switch back to HTTP to use the home proxy, but should be aware of the possibility of integrity attacks on non-signed data. In the case of ALTO home proxies running on ISP-managed gateways, the ISP may provide SSL/TLS certificates for each gateway, thus supporting SSL/TLS communications between clients and proxies. In this scenario, clients may reasonably assume that home proxies acting as local ALTO servers provides authoritative responses equivalent to those of the ISP's remote ALTO server.

As mentioned before, serving ALTO queries within the home reduces the amount of user-sensitive information that leaks to external parties (e.g., a remote ALTO server). Therefore, ALTO home proxies should enhance user privacy.

7 IANA Considerations

This document does not mandate any IANA actions.

8 References

8.1 Informative References

[RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

[RFC2608] Guttman, E., Perkins, C., Veizades, J., and M. Day, "Service Location Protocol, Version 2", RFC 2608, June 1999.

[I-D.gu-alto-redistribution] Yingjie, G., Alimi, R., and R. Even, "ALTO Information Redistribution", draft-gu-alto-redistribution-03 (work in progress), July 2010.

[I-D.ietf-alto-server-discovery] Kiesel, S., Siemerling, M., Schwan, N., Scharf, M., and H. Song, "ALTO Server Discovery", draft-ietf-alto-server-discovery-02 (work in progress), September 2011.

[I-D.ietf-alto-protocol] Alimi, R., Penno, R. and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-09 (work in progress), June 2011.

Authors' Addresses

Fabio Picconi
Technicolor
1 rue Jeanne d'Arc
92443 Issy les Moulineaux cedex
France

EMail: fabio.picconi@technicolor.com

Network Working Group S. Randriamasy, Ed.
Internet-Draft Alcatel-Lucent Bell Labs
Intended status: Experimental N. Schwan
Expires: January 12, 2012 July 11, 2011

Multi-Cost ALTO
draft-randriamasy-alto-multi-cost-03

Abstract

IETF is designing a new service called ALTO (Application Layer traffic Optimization) that includes a "Network Map Service", an "Endpoint Cost Service" and an "Endpoint (EP) Ranking Service" and thus incentives for application clients to connect to ISP preferred Endpoints. These services provide a view of the Network Provider (NP) topology to overlay clients.

The present draft proposes a light way to extend the information provided by the current ALTO protocol. The purpose is to broaden the possibilities of the Application Clients in two ways: firstly by providing a better mapping of the Selected Endpoints to needs of the growing diversity of Content Networking Applications and to the network conditions, secondly by producing a more robust choice of multiple Endpoints, helping thus out for efficient Multi-Path transfer.

There are 2 parts in this draft: the first part initiates protocol extensions to support requests on multiple Cost Types in one single transaction. These first extensions also integrate the discussions within the ALTO Working Group and focus on the Endpoint Cost Service. The second part proposes two use cases motivating further definitions of additional CostTypes and Cost Attributes related to timeframe and validity period.

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 [RFC2119].

Status of this Memo

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

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

working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on January 12, 2012.

Copyright Notice

Table of Contents

1. Introduction

IETF is designing a new service called ALTO that provides guidance to P2P applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource. This guidance shall be based on parameters that affect performance and efficiency of the data transmission between the hosts, e.g., the topological distance. The ultimate goal is to improve Quality of Experience (QoE) in the application while reducing resource consumption in the underlying network infrastructure. The ALTO protocol conveys the Internet View from the perspective of a Provider Network region that spans from a region to one or more Autonomous System (AS). Together with this Network Map, it provides the Provider determined Cost Map between locations of the Network Map. Last, it provides the Ranking of Endpoints w.r.t. their routing cost.

The term Network Provider in this document includes both ISPs, who provide means to transport the data and Content Delivery Network (CDN) operators who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.

The last ALTO protocol draft see [ID-alto-protocol], gives the possibility to query multiple Endpoint properties at once (see S.7.7.4.1). However section 7.7.3.2 on Cost Map states about both parameters Cost Type and Cost Mode that: "This parameter MUST NOT be specified multiple times". The ALTO requirements draft, see [ID-ALTO-Requirements7] also states in REQ. ARv05-14: "The ALTO client protocol MUST support the usage of several different rating criteria types". In the current protocol draft, there is no specified way to get values for several Cost Types altogether. Currently, the costs are provided in a scalar form, one by one. So that an ALTO Client wanting information for several Cost Types must place a request and receive a response as many times as desired Cost Types. However, vector costs provide a robust and natural input to multi-path connections and getting all costs in one single query/ response transaction saves time and ALTO traffic, thus ressources, thus energy.

The ALTO Problem Statement, see [RFC5693] and the ALTO requirements draft, see [RFC5693] stress that: "information that can change very rapidly, such as transport-layer congestion, is out of scope for an ALTO service. Such information is better suited to be transferred through an in-band technique at the transport layer instead", as "ALTO is not an admission control system "and does not necessarily know about the instant load of endpoints and links. However, longer term statistics or empirical ratings on performance oriented information may still be useful for a reliable choice of candidate endpoints. In addition, given the QoE requirements of nowadays and

future Internet applications, more and more NPs compute and store such information to optimize their traffic. Last, specific ALTO servers can be specified for mobile core networks, which have a smaller scale and can afford and take advantage of using smaller time-scale network information.

Adding QoE-enabling metrics to the Network Provider established routing cost could meet the interests of both the end users and the Providers. Besides, keeping the shortest or cheapest possible path, in addition, saves resources, time and energy.

2. Scope

This draft generalizes the case of a P2P client to include the case of a CDN client, a GRID application client and any Client having the choice in several connection points for data or resource exchange. To do so, it uses the term "Application Client" (AC).

This draft focuses on the use case where the ALTO client is embedded in the Application Client. For P2P applications, the use case where the ALTO Client is embedded in the P2P tracker is also applicable.

It is assumed that Applications likely to use the ALTO service have a choice in connection endpoints as it is the case for most of them. The ALTO service is managed by the Network Provider and reflects its preferences for the choice of endpoints. The NP defines in particular the network map, the routing cost among Network Locations, and which ALTO services are available at a given ALTO server.

The solution proposed in this draft is applicable to fixed networks. It is also meant for smaller networks such as mobile networks.

3. Terminology

Endpoint (EP): can be a Peer, a CDN storage location, a Party in a resource sharing swarm such as a computation Grid or an online multi-party game.

Endpoint Discovery (EP Discovery) : this term covers the different types of processes used to discover different types of endpoints.

Network Service Provider: includes both ISPs, who provide means to transport the data and Content Delivery Network (CDN) who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.

Application Client (AC): this term generalizes the case of a P2P client to include the case of a CDN client and of any Client having the choice in several connection points for data or resource exchange.

4. Proposed ALTO protocol updates for multi-cost transactions

This section is to be completed and proposes first updates of the ALTO protocol to support Multi Cost ALTO Services or provide additional ALTO information. It integrates discussions on he ALTO mailing list and its goal is to initiate further discussions and protocol update proposals.

If an ALTO client desires several Cost Types, instead of placing as many requests as costs, it may request and receive all the desired cost types in one single transaction. The correspondence between the components and the cost types MUST be indicated in the ALTO request.

The ALTO server then, provided it supports the desired cost, and provided it supports multi-cost ALTO transactions, sends one single response where for each {source, destination} pair. The cost values are arranged in a vector, whose component each corresponds to a specified Cost Type. The correspondence between the components and the cost types MUST be indicated either in the ALTO response or available via the resource directory.

The ALTO services impacted by the Multi-Cost extensions are:

- Information Resources Directory,
- Cost Map Service,
- Cost Map Filtering Service,
- Endpoint Cost Lookup Service.

This draft focuses on the case of the Endpoint Cost Lookup Service

4.1. Multi-Cost Map Service

To be completed

4.1.1. Media Type

4.1.2. HTTP Method

This resource is requested using the HTTP POST method

4.1.3. Input Parameters

4.1.4. Capabilities

4.1.5. Response

4.1.6. Example

4.2. Endpoint Multi-Cost Service

4.2.1. Media Type

4.2.2. HTTP Method

This resource is requested using the HTTP POST method

4.2.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies input parameters with a data format indicated by media type "application/alto-endpointcostparams+json", which is a JSON Object of type ReqEndpointCostMap:

object {

TypedEndpointAddr srcs<0..*>; [OPTIONAL]

TypedEndpointAddr dsts<1..*>;

} EndpointFilter;

object {

CostMode cost-mode;

CostType cost-type<1..*>;

JSONString constraints; [OPTIONAL]

EndpointFilter endpoints;

} ReqEndpointCostMap;

with members:

cost-mode The Cost Mode (Section 5.1.2) to use for returned costs. For Multi-Cost requests this Cost Mode SHOULD be numerical.

cost-type The Cost Type (Section 5.1.1) to use for returned costs. All the listed the Cost Types MUST be indicated in this resource's capabilities (Section 7.7.5.1.4).

constraints Defined equivalently to the "constraints" input parameter of a Filtered Cost Map (see Section 7.7.3.2).

endpoints A list of Source Endpoints and Destination Endpoints for which Path Costs are to be returned. If the list of Source Endpoints is empty (or not included), the ALTO Server MUST interpret it as if it contained the Endpoint Address corresponding Alimi, et al. Expires December 29, 2011 [Page 50] Internet-Draft ALTO Protocol June 2011 to the client IP address from the incoming connection (see Section 10.3 for discussion and considerations regarding this mode). The list of destination Endpoints MUST NOT be empty. The ALTO Server MUST interpret entries appearing multiple times in a list as if they appeared only once.

4.2.4. Capabilities

4.2.5. Response

4.2.6. Example

4.3. ALTO Status Codes for Multi-Cost ALTO

If the Multi-cost Service is not supported for either the Cost Map or the Endpoint Service, then the ALTO server sends an ALTO status code 7 corresponding to HTTP status code 501 indicating "Invalid cost structure". The ALTO client may then needs to place as many requests as needed Cost Types, and the ALTO server sends as many cost maps or EP cost as needed.

To the attribute Cost Mode in S.5.1 should be associated a rule stipulating that when multiple cost types are requested, then the requested Cost Mode SHOULD be numerical.

5. Use cases for further Cost Types and Endpoint Properties

The current ALTO protocol [ID-alto-protocol] specification requests the creation of two registries maintained by IANA. The ALTO Cost Type registry ensures that the Cost Types that are represented by an ALTO Cost Map are unique identifiers, and it further contains references to the semantics of the Cost Type. The current

specification registers 'routingcost' as a generic measure for routing traffic from a source to a destination. In a similar way the ALTO Endpoint Property Registry ensures uniquenes of ALTO Endpoint Property identifiers and provides references to particular semantics of the allocated Enpoint Properties. Currently the 'pid' identifier is registered, which serves as an identifier that allows aggregation of network endpoints into network regions. Both registries accept new entries after Expert Review [[ID-alto-protocol]]. New entries are requested to be conform to the respective syntactical requirements, and must include information about the new identifier, the intended semantics as well as security considerations.

The current protocol specification concentrates on the basic use case of optimizing routing costs in NSPs networks. Upcoming use cases however will require both, new Cost Types and new Endpoint Properties. The goal of this section is to describe further forward looking use case scenarios that are likely to benefit from ALTO, and, in future iterations, to convey new Cost Types and Endpoint Properties that are likely to be beneficial for ALTO clients in these scenarios.

5.1. Delay Sensitive Overlay Applications

The ALTO working group has been created to allow P2P applications and NSPs a mutual cooperation, in particular because P2P bulk file-transfer applications have created a huge amount of intra-domain traffic. By aligning overlay topologies according to the 'routingcost' of the underlying network both layers are expected to benefit in terms of reduced costs and improved Quality-of-Experience.

However other types of overlay applications might benefit from a different set of path metrics. In particular for real-time sensitive applications, such as gaming, interactive video conferencing or medical services, creating an overlay topology with respect to a minimized delay is preferable. However it is very hard for a NSP to give accurate guidance for this kind of realtime information, instead probing through end-to-end measurements on the application layer has proven to be the superior mechanism. Still, a NSP might give some guidance to the overlay application, for example by providing statisically preferable paths with respect to the time of a day. Also static information like hopcount can be seen as an indicator for the delay that can be expected. In the following iterations this draft will thus analyse which metrics can realisticaly be provided through ALTO to give delay sensitive applications guidance for peer selection.

5.2. CDN Surrogate Selection

A second use case is motivated through draft [draft-jenkins-alto-cdn-use-cases-01]. The request router in today's CDNs makes a decision about to which surrogate or cache node a content request should be forwarded to. Typically this decision is based on locality aspects, i.e. the surrogate node which is closest to the client is preferred by the request router. An ALTO server hereby is one promising option to allow NSPs to give guidance to the CDN about which cache node would be preferable according to the view of the network by the 'routingcost' Cost Type. Providing this kind of information is in particular important as one trend is to place cache nodes deeper into the network, which results in the need for finer grained information.

While distance today is the predominant metric used for routing decisions, other metrics might allow sophisticated request routing strategies. For example the load a cache node sees in terms of CPU utilization, memory usage or bandwidth utilization might influence routing decisions for load-balancing reasons. There exist numerous ways of gathering and feeding this kind of information into the request routing mechanism. As ALTO is likely to become a standardized interface to provide network topology information, for simplicity other information that is used by a request router could be provided by the ALTO server as well. In the next iterations of this draft we will analyse which of these metrics is suitable to be provided as Cost Type or Endpoint Property for the use case of CDN Surrogate Selection and propose to register them in the respective registries.

6. Proposed additional Properties and Costs

To be further specified

6.1. Scoping ALTO information

One way for the NSP to provide guidance on highly dynamic network state information such as delay and load is to provide them in the form of statistics or as a numerical coarse grain indicator. It is important to have the possibility to reflect that the provided values are applicable for a given time period, for example business hours, and are subject to changes over time.

The following attributes can be associated to the applicable ALTO information:

- an age attribute indicating when the information was generated.
- for statistical costs a time period attribute indicating over which duration the statistics were collected.
- a validity attribute indicating when the provided values should be refreshed. By defaut, this parameter can be set to infinity.

7. IANA Considerations

Information for the ALTO Endpoint property registry maintained by the IANA and related to the new Endpoints supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Endpoint Property Registry" of [ID-alto-protocol],

Information for the ALTO Cost Type Registry maintained by the IANA and related to the new Cost Types supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Cost Type Registry" of [ID-alto-protocol],

7.1. Information for IANA on proposed Cost Types

When a new ALTO Cost Type is defined, accepted by the ALTO working group and requests for IANA registration MUST include the following information, detailed in Section 11.2: Identifier, Intended Semantics, Security Considerations.

7.2. Information for IANA on proposed Endpoint Propeeries

Likewise, an ALTO Endpoint Property Registry could serve the same purposes as the ALTO Cost Type registry. Application to IANA registration for Endpoint Properties would follow a similar process.

8. Acknowledgements

Sabine Randriamasy is partially supported by the MEDIEVAL project (http://www.ict-medieval.eu/), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the MEDIEVAL project or the European Commission.

Nico Schwan is partially supported by the ENVISION project

(http://www.envision-project.org), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ENVISION project or the European Commission.

9. References

9.1. Normative References

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

[RFC5693] "Application Layer Traffic Optimization (ALTO) Problem Statement", October 2009.

9.2. Informative References

[ID-ALTO-Requirements7] "draft-ietf-alto-reqs-07.txt", January 2011.

[ID-alto-protocol] , Eds., ""ALTO Protocol" draft-ietf-alto-protocol-09.txt", June 2011.

[draft-jenkins-alto-cdn-use-cases-01] ""Use Cases for ALTO within CDNs" draft-jenkins-alto-cdn-use-cases-01", June 2011.

Authors' Addresses

Sabine Randriamasy (editor)
Alcatel-Lucent Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

Nico Schwan

Phone:
Fax:
Email:
URI:

Network Working Group S. Randriamasy, Ed.
Internet-Draft N. Schwan
Intended status: Experimental Alcatel-Lucent Bell Labs
Expires: May 3, 2012 October 31, 2011

Multi-Cost ALTO
draft-randriamasy-alto-multi-cost-05

Abstract

IETF is designing a new service called ALTO (Application Layer traffic Optimization) that includes a "Network Map Service", an "Endpoint Cost Service" and an "Endpoint (EP) Ranking Service" and thus incentives for application clients to connect to ISP preferred Endpoints. These services provide a view of the Network Provider (NP) topology to overlay clients.

The present draft proposes a light way to extend the information provided by the current ALTO protocol. The purpose is to broaden the possibilities of the Application Clients in two ways. Firstly it proposes to include information on multiple cost types in a single ALTO transaction, providing a better mapping of the Selected Endpoints to needs of the growing diversity of Content Networking Applications and to the network conditions. Secondly it proposes new cost types, that are an abstraction of time-sensitive network information as viewed by the Network Provider. All this also helps producing a more robust choice when multiple Endpoints must be selected.

There are 2 parts in this draft: the first part initiates protocol extensions to support requests on multiple Cost Types in one single transaction. These first extensions also integrate the discussions within the ALTO Working Group. The second part proposes use cases motivating further definitions of additional CostTypes and Cost related attributes and capabilities.

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 [RFC2119].

Status of this Memo

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

This Internet-Draft will expire on May 3, 2012.

Copyright Notice

Table of Contents

1. Introduction

IETF is designing a new service called ALTO that provides guidance to P2P applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource. This guidance shall be based on parameters that affect performance and efficiency of the data transmission between the hosts, e.g., the topological distance. The ultimate goal is to improve Quality of Experience (QoE) in the application while reducing resource consumption in the underlying network infrastructure. The ALTO protocol conveys the Internet View from the perspective of a Provider Network region that spans from a region to one or more Autonomous System (AS). Together with this Network Map, it provides the Provider determined Cost Map between locations of the Network Map. Last, it provides the Ranking of Endpoints w.r.t. their routing cost.

The term Network Provider in this document includes both ISPs, who provide means to transport the data and Content Delivery Network (CDN) operators who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.

The last ALTO protocol draft see [ID-alto-protocol], gives the possibility to query multiple Endpoint properties at once. On the other hand, the Endpoint Cost service in its input specification allows only one Cost Type and Cost mode per request. The ALTO requirements draft, see [ID-ALTO-Requirements7] states in REQ. ARv05-14: "The ALTO client protocol MUST support the usage of several different rating criteria types". In the current protocol draft, there is no specified way to get values for several Cost Types simultaneously. Currently, the costs are provided in a scalar form, one by one. So that an ALTO Client wanting information for several Cost Types must request and receive a response as many times as desired Cost Types.

Getting all costs in one single query/response transaction saves time and ALTO traffic load, thus ressources, thus energy. Besides, vector costs provide a robust and natural input to multi-variate path computation as well as robust multi-variate selection multiple Endpoints. Other savings in resources can be obtained by gathering multiple Cost Types in the Cost map and Filtered Cost Map services. Indeed, one Cost Map reporting on N Cost Types is less bulky than N Cost Maps containing one Cost Type each. This is valuable for both the storage of these maps and their transfer. Last, as it is most likely that nowadays applications need information on several cost types, having them gathered in one map will save time.

The ALTO Problem Statement, see [RFC5693] and the ALTO requirements draft, see [RFC5693] stress that: "information that can change very

rapidly, such as transport-layer congestion, is out of scope for an ALTO service. Such information is better suited to be transferred through an in-band technique at the transport layer instead", as "ALTO is not an admission control system "and does not necessarily know about the instant load of endpoints and links. However, non-real time abstraction of performance oriented information is useful for a reliable choice of candidate endpoints. In addition, given the QoE requirements of nowadays and future Internet applications, more and more NPs compute and store such information to optimize their traffic. Besides specific ALTO servers can be specified for small networks including mobile core networks, which have a smaller scale and can afford and take advantage of using small time-scale network information. Adding QoE-enabling metrics to the Network Provider established routing cost could meet the interests of both the end users and the Providers.

2. Scope

This draft generalizes the case of a P2P client to include the case of a CDN client, a GRID application client and any Client having the choice in several connection points for data or resource exchange. To do so, it uses the term "Application Client" (AC).

This draft focuses on the use case where the ALTO client is embedded in the Application Client or in some Application Endpoint tracker in the network, such as a P2P tracker, a CDN location tracker or a cloud computing orchestration system implemented in a logically centralized management system.

It is assumed that Applications likely to use the ALTO service have a choice in connection endpoints as it is the case for most of them. The ALTO service is managed by the Network Provider and reflects its preferences for the choice of endpoints. The NP defines in particular the network map, the routing cost among Network Locations, and which ALTO services are available at a given ALTO server.

3. Terminology

Endpoint (EP): can be a Peer, a CDN storage location, a Party in a resource sharing swarm such as a computation Grid or an online multi-party game.

Endpoint Discovery (EP Discovery) : this term covers the different types of processes used to discover different types of endpoints.

Network Service Provider (NSP): includes both ISPs, who provide means

to transport the data and Content Delivery Network (CDN) who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.

ALTO transaction: a request/response exchange between an ALTO Client and an ALTO Server.

Application Client (AC): this term generalizes the case of a P2P client to include the case of a CDN client and of any Client having the choice in several connection points for data or resource exchange.

4. Proposed ALTO protocol updates for multi-cost transactions

This section proposes updates of the ALTO protocol to support Multi Cost ALTO Services or provide additional ALTO information. It integrates discussions on the ALTO mailing list.

If an ALTO client desires several Cost Types, instead of placing as many requests as costs, it may request and receive all the desired cost types in one single transaction.

The ALTO server then, provided it supports the requested Cost Types, and provided it supports multi-cost ALTO transactions, sends one single response where for each {source, destination} pair, the cost values are arranged in a vector, where each component corresponds to a specified Cost Type. The correspondence between the components and the cost types MUST be indicated either in the ALTO response or available via the resource directory.

The following ALTO services get corresponding services with Multi-Cost extensions:

- Information Resources Directory: extended with multi-cost related URIs and associated capabilities.
- Cost Map Service: extended with the Multi-Cost Map Service,
- Cost Map Filtering Service: extended with the Multi-Cost Map Filtering Service,
- Endpoint Cost Lookup Service: extended with the Endpoint Multi-Cost Lookup Service.

4.1. Information Resources Directory

When the ALTO server supports the provision of information on multiple costs in a single transactions, the Information Resources will list the corresponding resources. The media type and encoding specificarions remain the same as in the current ALTO protocol.

4.1.1. Example Multi-Cost specific resources

The following is an example Information Resource Directory returned by an ALTO Server and containing Multi-Cost specific services: the Multi-Cost Map Service, Filtered Multi-Cost Map and the Endpoint Multi-Cost Service. It is assumed that the IRD contains usual ALTO Services as described in the example IRD of the current ALTO protocol. In this example, the ALTO Server provides additional Multi-Cost Services in a specific folder of "alto.example.com" called "multi". This folder contains the Multi-Cost Maps, Filtered Multi-Cost Maps as well as the Endpoint Multi-Cost Service.

```
GET /directory HTTP/1.1
   Host: alto.example.com
   Accept: application/alto-directory+json,application/alto-error+json



   HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-directory+json

   {
     "resources" : [
       {
         .....
         Usual ALTO "single-cost" Services as described in ALTO Protocol
         .....
       }, {
         "uri" : "http://alto.example.com/multi/maps",
         "media-types" : ["application/alto-multicostmap+json"],
         "accepts" : ["application/alto-multicostmapfilter+json"],
         "capabilities" : {
           "cost-constraints" : true,
           "cost-types" : [ "routingcost", "hopcount" ],
           "cost-modes" : [ "numerical", "numerical" ]
         }
       }, {
         "uri" : "http://alto.example.com/multi/endpointmulticost/lookup",
         "media-types" : [ "application/alto-endpointmulticost+json" ],
         "accepts" : [ "application/alto-endpointmulticostparams+json" ],
         "capabilities" : {
           "cost-constraints" : true,
           "cost-types" : [ "routingcost", "hopcount" ],
           "cost-modes" : [ "numerical", "numerical" ]
         }
       }, {
         "uri" : "http://custom.alto.example.com/multi/endpointmulticost/lookup",
         "media-types" : [ "application/alto-endpointmulticost+json" ],
         "accepts" : [ "application/alto-endpointmulticostparams+json" ],
       }
     ]
   }
```

4.2. Multi-Cost Map Service

This section introduces a new media-type for the Multi-Cost map. For each source/destination pair of PIDs, it provides the value of the different Cost Type supported for the Multi-cost map, in the same order as in the list of cost-types specified in the capabilities.

A Multi-Cost Map MAY be provided by an ALTO Server.

This resource MUST be provided for at least the 'routingcost' Cost Type with the 'numerical' Cost Mode. It is assumed that an ALTO Server supporting multi-cost maps supports the 'numerical' Cost Mode for all Cost Types encoded in the 'JSONnumber' type.

Note that the capabilities specify implicitly the order in which the different Cost Type values will be listed in the Cost Map.

The Cost Type values in the responses are encoded in with a JSONArray of cost values for the different required cost types.

Note also that contrary to the Cost Map service, the returned Multi Cost Map is not required to include the required Path Costs for each pair of Source and Destination PID known to the ALTO Server. The reason is that for a given source/destination pair, the ALTO Server may not have the information on certain Cost Types. As a consequence, contrary to the Cost Map service, the Multi-Cost Map service introduces a particular value that unambiguously indicates that the information is not available. This way, the order in which the cost values are provided for a source/destination pair is unambiguous.

4.2.1. Media Type

The media type is "application/alto-multicostmap+json".

4.2.2. HTTP Method

This resource is requested using the HTTP GET method.

4.2.3. Input Parameters

None.

4.2.4. Capabilities

This resource may be defined for multiple Cost Types and Cost Modes. The capabilities of an ALTO Server URI providing this resource are defined by a JSON Object of type MultiCostMapCapability:

```
object {
     CostType cost-types<1..*>;
     CostMode cost-modes<0..*>;
   } MultiCostMapCapability;
```

with members

cost-types The Cost Types (Section 5.1.1) supported by the corresponding URI. This member MUST at least include the type 'routingcost'. The order in which the Cost Type values for a source/destination pair will be listed in the Multi-Cost Map provided to an ALTO Client MUST be the order in which these Cost Types are listed in this member.

cost-modes The Cost Mode (Section 5.1.2) supported for each of the supported Cost Types listed in the "cost-types". For Cost types encoded with the 'JSONnumber' type, the Cost Mode MUST be numerical. It will be interpreted as such if this member is not present.

An ALTO Server MUST support all of the Cost Types listed here for each of the listed Cost Modes. Note that an ALTO Server may provide multiple Cost Map Information Resources, each with different capabilities.

An ALTO Server supporting the Multi-Cost Map service, MUST support the Cost mode 'numerical' for all supported Cost Types encoded with the 'JSONnumber' type. It also MUST list the Cost Type values associated to a source/destination pair in the same order as in the "cost-types" member of the capabilities specified the Multi-Cost Map resource.

4.2.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceMultiCostMap:

```
object DstMultiCosts {
     JSONArray [PIDName];
     ...
   };

   object {
     DstMultiCosts [PIDName]<0..*>;
     ...
   } MultiCostMapData;

   object {
     CostType    cost-type<1..*>;
     CostMode    cost-mode<1..*>;
     JSONString  map-vtag;
     MultiCostMapData map;
   } InfoResourceMultiCostMap;
```

with members:

cost-mode Cost Mode (Section 5.1.2) used in the Cost Map where each member of the cost-mode list is the Cost Mode provided for the Cost Type at the same place in the list.

cost-type Cost Type (Section 5.1.1) used in the Multi Cost Map.

map-vtag The Version Tag (Section 5.3) of the Network Map used to generate the Cost Map.

map The Multi Cost Map data itself.

MultiCostMapData is a JSON object with each member representing a single Source PID; the name for a member is the PIDName string identifying the corresponding Source PID. For each Source PID, a DstMultiCosts object denotes the associated multiple costs to a set of destination PIDs (Section 5.2); the name for each member in the object is the PIDName string identifying the corresponding Destination PID. DstMultiCosts are listed in the same order as in the 'cost-type' array.

The returned Cost Map MUST include the required Path Costs for each pair of Source and Destination PID for which this information is available.

The members cost-mode and cost-type MUST be arrays with the same number of elements.

Note also that the Multi-Cost Map service needs a particular value that unambiguously indicates that the information is not available.

As an example this value is referred here to as NAv for "Not available". Note that the type of NAv still needs to be specified: preferably a numerical value for numerical costs that unambiguously means: "not available" and can distiguished from "infinite" or "invalid something" or any "pathological" value.

4.2.6. Example

This example illustrates a 'static' multi-cost' ALTO trasaction, where the utilized cost-types all have 'static' values. We assume here that the Cost Types available at the ALTO Server are "routingcost" and "hopcount" and the 'numerical' mode is available for both of them. The "routingcost" may be based on monetary considerations where as the "hopcount" is used to account on the path delay.

```
GET /multicostmap/num HTTP/1.1
   Host: alto.example.com
   Accept: application/alto-multicostmap+json,application/alto-error+json

HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-multicostmap+json

   {
     "meta" : {},
     "data" : {
       "cost-mode" : ["numerical", "numerical"]
       "cost-type" : ["routingcost", "hopcount"]
       "map-vtag"  : "1266506139",
       "map" : {
         "PID1": { "PID1": [1,6],  "PID2": [5,23],  "PID3": [10,5] },
         "PID2": { "PID1": [5,5],  "PID2": [1,11],  "PID3": [15,9] },
         "PID3": { "PID1": [20,12], "PID2": [15,1], "PID3": [1,18]  }
       }
     }
   }
```

4.3. Filtered Multi-Cost Map

A Multi-Cost Map may reach a huge volume and also, an Application Client assisted by the ALTO Client does not necessarily need information on all the Cost Types for all the source/destination pairs of PIDs.

Therefore, applications may more likely use Cost Map information filtered w.r.t. the Cost types as well as the source/destination

pairs of PIDs. This section specifies filtered Multi-Cost Maps.

A Filtered Cost Map is a Cost Map Information Resource (Section 7.7.2.2) for which an ALTO Client may supply additional parameters limiting the scope of the resulting Cost Map. A Filtered Cost Map MAY be provided by an ALTO Server.

4.3.1. Media Type

The media type is "application/alto-multicostmap+json", see Section 4.2.1 of this draft.

4.3.2. HTTP Method

This resource is requested using the HTTP POST method

4.3.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/ alto-multicostmapfilter+json", which is a JSON Object of type ReqFilteredCostMap, where:

```
object {
     PIDName srcs<0..*>;
     PIDName dsts<0..*>;
   } PIDFilter;

   object {
     CostType    cost-type<0..*>;
     CostMode    cost-mode<0..*>;
     JSONString constraints<0..*>;   [OPTIONAL]
     PIDFilter  pids;                [OPTIONAL]
   } ReqFilteredMultiCostMap;
```

with members:

cost-type The Cost Type (Section 5.1.1) for the returned costs. Each listed cost-type MUST be one of the supported Cost Types indicated in this resource's capabilities (Section 7.7.3.2.4).

cost-mode The Cost Mode (Section 5.1.2) for each of the returned cost-types. For Cost types encoded with the 'JSONnumber' type, the Cost Mode MUST be numerical. It will be interpreted as such if this member is not present.

constraints Defines a list of additional constraints on which elements of the Cost Map are returned. This parameter MUST NOT be specified if this resource's capabilities (Section 7.7.3.2.4) indicate that constraint support is not available. A constraint contains two entities separated by whitespace: (1) an operator either 'gt' for greater than or 'lt' for less than (2) a target numerical cost. The numerical cost is a number that MUST be defined in the same units as the Cost Type indicated by the cost-type parameter. ALTO Servers SHOULD use at least IEEE 754 double-precision floating point [IEEE.754.2008] to store the numerical cost, and SHOULD perform internal computations using double-precision floating-point arithmetic. If multiple 'constraint' parameters are specified, they are interpreted as being related to each other with a logical AND.

pids A list of Source PIDs and a list of Destination PIDs for which Path Costs are to be returned. If a list is empty, the ALTO Server MUST interpret it as the full set of currently-defined PIDs. The ALTO Server MUST interpret entries appearing in a list multiple times as if they appeared only once. If the "pids" member is not present, both lists MUST be interpreted by the ALTO Server as containing the full set of currently-defined PIDs.

4.3.4. Capabilities

The URI providing this resource supports all capabilities documented in Section 7.7.2.2.4 (with identical semantics), plus additional capabilities. In particular, the capabilities are defined by a JSON object of type FilteredMultiCostMapCapability:

```
object {
     CostMode cost-modes<0..*>;
     CostType cost-types<0..*>;
     JSONBool cost-constraints;
   } FilteredMultiCostMapCapability;
```

with members:

cost-modes See Section 4.2.5 of this MC draft

cost-types See Section 4.2.5 of this MC draft

cost-constraints If true, then the ALTO Server allows cost constraints to be included in requests to the corresponding URI. If not present, this member MUST be interpreted as if it specified false.

4.3.5. Response

See Section of this draft for the format. The returned Cost Map MUST NOT contain any source/destination pair that was not indicated (implicitly or explicitly) in the input parameters. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as if the PID did not appear in the input parameters. If any constraints are specified, Source/ Destination pairs that do for which the Path Costs do not meet the constraints MUST NOT be included in the returned Cost Map. If no constraints were specified, then all Path Costs are assumed to meet the constraints.

4.3.6. Example

```
POST multi/multicostmap/filtered HTTP/1.1
   Host: alto.example.com
   Content-Type: application/alto-multicostmapfilter+json
   Accept: application/alto-multicostmap+json,application/alto-error+json

   {
     "cost-mode" : "numerical", "numerical"],
     "cost-type" : "routingcost", "hopcount"],
     "pids" : {
       "srcs" : [ "PID1" ],
       "dsts" : [ "PID1", "PID2", "PID3" ]
     }
   }


   HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-costmap+json

   {
     "meta" : {},
     "data" : {
       "cost-mode" : ["numerical", "numerical"],
       "cost-type" : ["routingcost", "hopcount"],
       "map-vtag" : "1266506139",
       "map" : {
         "PID1": { "PID1": [1,6],  "PID2": [5,23],  "PID3": [10,5] }
       }
     }
   }
```

4.4. Endpoint Multi-Cost Service

The Endpoint Multi-Cost Service provides information about several costs between individual Endpoints.

This service does not allow lists of Endpoint prefixes (and addresses, as a special case) to be ranked (ordered) by an ALTO Server, as firstly the costs encoded with the JSONnumber 'type' are provided in the numerical Mode and secondly the choice among various existing methods to rank Endpoints upon multiple costs (criteria) is out of scope of this protocol and in the responsability of the Application Client using the ALTO Endpoint Multi-Cost information.

However common sense lead to warn that a necessary condition for methods that rank vectors to be reliable is that the components (costs) of the processed vectors be numerical Cost Mode.

This Service introduces a new media type to define the service and the input parameters.

4.4.1. Endpoint Multi-Cost

The Endpoint Multi-Cost resource provides information about multiple costs between individual endpoints.

This service MAY be provided by an ALTO Server. If it is provided. It is important to note that although this resource allows an ALTO Server to reveal costs between individual endpoints, an ALTO Server is not required to do so. A simple alternative would be to compute the cost between two endpoints as the cost between the PIDs corresponding to the endpoints +++ if this service is available for the requested Cost Types +++ . See Section 12.1 for additional details.

4.4.2. Media Type

The media type is "application/alto-endpointmulticost+json".

4.4.3. HTTP Method

This resource is requested using the HTTP POST method

4.4.4. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies input parameters with a data format indicated by media type "application/alto-endpointmulticostparams+json", which is a JSON Object of type ReqEndpointMultiCostMap:

```
object {
        TypedEndpointAddr srcs<0..*>; [OPTIONAL]
        TypedEndpointAddr dsts<1..*>;
 } EndpointFilter;

 object{
       CostType cost-type<0..*>;
       CostMode cost-mode<0..*>;
       JSONString constraints; [OPTIONAL]
       EndpointFilter endpoints;
 } ReqEndpointMultiCostMap;
```

with members:

cost-mode The Cost Mode (Section 5.1.2) to use for returne costs that are encoded with the 'JSONnumber' type. For Multi-Cost requests this Cost Mode MUST be numerical for any Cost Type encoded with the 'JSONnumber' type, provided that the Cost Mode 'numerical' is available for this Cost Type. Remember (Section 5.1.2) that ALTO Clients SHOULD be cognizant of operations applicable to different Cost Modes.

cost-type The Cost Type (Section 5.1.1) to use for returned costs. All the listed the Cost Types MUST be indicated in this resource's capabilities (Section 7.7.5.1.4).

constraints Defined equivalently to the "constraints" input parameter of a Filtered Multi Cost Map (see Section 7.7.3.2).

endpoints A list of Source Endpoints and Destination Endpoints for which Path Costs are to be returned. If the list of Source Endpoints is empty (or not included), the ALTO Server MUST interpret it as if it contained the Endpoint Address corresponding to the client IP address from the incoming connection (see Section 10.3 for discussion and considerations regarding this mode). The list of destination Endpoints MUST NOT be empty. The ALTO Server MUST interpret entries appearing multiple times in a list as if they appeared only once.

4.4.5. Capabilities

See section 4.3.4 of this draft.

4.4.6. Response

The returned InfoResourceEntity object has "data" member equal to InfoResourceEndpointMultiCostMap, where:

```
object EndpointDstMultiCosts {
     JSONArray [TypedEndpointAddr];
     ...
   };

   object {
     EndpointDstMultiCosts [TypedEndpointAddr]<0..*>;
     ...
   } EndpointMultiCostMapData;

   object {
     CostMode            cost-mode<0..*>;
     CostType            cost-type<0..*>;
     EndpointMultiCostMapData map;
   } InfoResourceEndpointMultiCostMap;
```

InfoResourceEndpointMultiCostMap has members:

cost-type<0..*> The Cost Types used in the returned Cost Map.

cost-mode<0..*> The Cost Mode for each of the Cost Types used in the returned Cost Map.

map The Endpoint Multi-Cost Map data itself.

EndpointMultiCostMapData is a JSON object with each member representing a single Source Endpoint specified in the input parameters; the name for a member is the TypedEndpointAddr string identifying the corresponding Source Endpoint. For each Source Endpoint, a EndpointDstMultiCosts object denotes the cost vector associated to each Destination Endpoint specified in the input parameters; the name for each member in the object is the TypedEndpointAddr string identifying the corresponding Destination Endpoint.

4.4.7. Example

```
POST /endpointcost/lookup HTTP/1.1
  Host: alto.example.com
  Content-Length: [TODO]
  Content-Type: application/alto-endpointmulticostparams+json
  Accept: application/alto-endpointmulticost+json,application/alto-error+json

  {
    "cost-type" : ["routingcost", "hopcount"],
    "cost-mode" : ["numerical", "numerical"],
    "endpoints" : {
      "srcs": [ "ipv4:192.0.2.2" ],
      "dsts": [
        "ipv4:192.0.2.89",
        "ipv4:198.51.100.34",
        "ipv4:203.0.113.45"
      ]
    }
  }


  HTTP/1.1 200 OK
  Content-Length: [TODO]
  Content-Type: application/alto-endpointmulticost+json

  {
    "meta" : {},
    "data" : {
      "cost-type" : ["routingcost", "hopcount"],
      "cost-mode" : ["numerical", "numerical"],
      "map" : {
        "ipv4:192.0.2.2": {
          "ipv4:192.0.2.89"    : [1, 7],
          "ipv4:198.51.100.34" : [2, 4],
          "ipv4:203.0.113.45"  : [3, 2]
        }
      }
    }
  }
```

4.5. ALTO Status Codes for Multi-Cost ALTO

If the Multi-cost Service is not supported for either the Cost Map or the Endpoint Service, then the ALTO server sends an ALTO status code 7 corresponding to HTTP status code 501 indicating "Invalid cost structure". The ALTO client may then needs to place as many requests as needed Cost Types, and the ALTO server sends as many cost maps or

EP cost as needed.

To the attribute Cost Mode in S.5.1 should be associated a rule stipulating that when multiple cost types are requested, then the requested Cost Mode SHOULD be numerical.

5. Use cases for further Cost Types and Endpoint Properties

The current ALTO protocol [ID-alto-protocol] specification requests the creation of two registries maintained by IANA. The ALTO Cost Type registry ensures that the Cost Types that are represented by an ALTO Cost Map are unique identifiers, and it further contains references to the semantics of the Cost Type. The current specification registers 'routingcost' as a generic measure for routing traffic from a source to a destination. In a similar way the ALTO Endpoint Property Registry ensures uniqueness of ALTO Endpoint Property identifiers and provides references to particular semantics of the allocated Enpoint Properties. Currently the 'pid' identifier is registered, which serves as an identifier that allows aggregation of network endpoints into network regions. Both registries accept new entries after Expert Review [[ID-alto-protocol]]. New entries are requested to be conform to the respective syntactical requirements, and must include information about the new identifier, the intended semantics as well as security considerations.

The current protocol specification concentrates on the basic use case of optimizing routing costs in NSPs networks. Upcoming use cases however will require both, new Cost Types and new Endpoint Properties. The goal of this section is to describe further forward looking use case scenarios that are likely to benefit from ALTO, and, in future iterations, to convey new Cost Types and Endpoint Properties that are likely to be beneficial for ALTO clients in these scenarios.

5.1. Delay Sensitive Overlay Applications

The ALTO working group has been created to allow P2P applications and NSPs a mutual cooperation, in particular because P2P bulk file-transfer applications have created a huge amount of intra-domain and congestion on low-speed uplink traffic. By aligning overlay topologies according to the 'routingcost' of the underlying network both layers are expected to benefit in terms of reduced costs and improved Quality-of-Experience.

However other types of overlay applications might benefit from a different set of path metrics. In particular for real-time sensitive applications, such as gaming, interactive video conferencing or

medical services, creating an overlay topology with respect to a minimized delay is preferable. However it is very hard for a NSP to give accurate guidance for this kind of realtime information, instead probing through end-to-end measurements on the application layer has proven to be the superior mechanism. Still, a NSP might give some guidance to the overlay application, for example by providing statistically preferable paths with respect to the time of a day. Also static information like hopcount can be seen as an indicator for the delay that can be expected. In the following iterations this draft will thus analyse which metrics can realistically be provided through ALTO to give delay sensitive applications guidance for peer selection.

5.2. CDN Surrogate Selection

A second use case is motivated through draft [draft-jenkins-alto-cdn-use-cases-01]. The request router in today's CDNs makes a decision about to which surrogate or cache node a content request should be forwarded to. Typically this decision is based on locality aspects, i.e. the surrogate node which is closest to the client is preferred by the request router. An ALTO server hereby is one promising option to allow NSPs to give guidance to the CDN about which cache node would be preferable according to the view of the network by the 'routingcost' Cost Type. Providing this kind of information is in particular important as one trend is to place cache nodes deeper into the network (i.e., closer to the end user), which results in the need for finer grained information.

While distance today is the predominant metric used for routing decisions, other metrics might allow sophisticated request routing strategies. For example the load a cache node sees in terms of CPU utilization, memory usage or bandwidth utilization might influence routing decisions for load-balancing reasons. There exist numerous ways of gathering and feeding this kind of information into the request routing mechanism.

Typically, information reporting on the occupation of a cache could be based on:

- an Endpoint Property called : "EPCapacity" and reflecting the nominal capacity of this endpoint. This capacity could be splitted in:
  - EP Nominal Memory : denoting the nominal storage capacity
  - EP Nominal Bandwidth: denoting the computation resources of the Endpoint.

- an Endpoint Cost called: "EP occupied Capacity" and reflecting the currently available resources wrt their nominal capacity and splitted in the same way as for the EP Capacity:
  - EP Occupied Memory: denoting the remaining storage capacity,
  - EP Occupied Bandwidth: denoting the remaining computation resources.

As ALTO is likely to become a standardized interface to provide network topology information, for simplicity other information that is used by a request router could be provided by the ALTO server as well. In the next iterations of this draft we will analyse which of these metrics is suitable to be provided as Cost Type or Endpoint Property for the use case of CDN Surrogate Selection and propose to register them in the respective registries.

5.3. Bulk Data Transfer scheduling

Applications like Facebook or YouTube rely on data replication across multiple sites for several reasons, such as offloading the core network or increasing user experience through short latency.

As content is generated constantly data also needs to be replicated across the various locations of a CDN provider, leading to bulk data transfers between datacenters. Scheduling these data transfers is a non-trivial task as the transfer should not infer with the user peak demand to avoid degradation of user experience and to decrease billing costs for the datacenter operator by leveraging off-peak hours for the transfer. This peak demand typically follows a diurnal pattern according to the geographic region of the datacenter. One precondition to schedule transfers however is to have a good knowledge about the demand and link utilization patterns between the different datacenters and networks. Provisioning this data gets increasingly complex with the number of CDN nodes and in particular the number of datacenter operators that are involved. ALTO can provide time sensitive utilization maps through a dedicated service to allow CDN operators a mutual scheduling of such data transfers across administrative domains, which becomes even more significant through the CDNi protocol.

Another use case that stresses the need for multi-timeframe information is the one of private users or user groups having limitations in their connectivity either in time or resources or both. These kind of users definitely need to plan their data transfers to and from various locations an be sure to optimize for example the 'routingcost' jointly with some 'pathoccupationcost' and possibly the 'hopcount'. These users often have very poor means to

have any information on the network that may provide them some guidance and the ALTO protocol could greatly help them while optimizing traffic on networks that face continuous resources and/or connectivity challenges.

In the next iteration of this draft we plan specify a multi-timeframe map service for ALTO to allow the targeted scheduling of data transfers between datacenter locations or other types of locations.

6. Proposed additional Properties and Costs

This section proposes further extensions on new Costs Types, to be discussed in the WG.

6.1. For further extensions: dynamic Costs

It is agreed in the ALTO requirements that: "information that can change very rapidly, such as transport-layer congestion, is out of scope for an ALTO service. Such information is better suited to be transferred through an in-band technique at the transport layer instead". However NP managing ALTO Servers as well as Application Client using ALTO information have common interests to use some information on time (or space) varying information that is not provided in real time, which is neither desirable nor feasible, but rather a synthetis of such information. Such information is made available for ALTO Servers in order to reflect how the NP wishes the information to be used by the Applicaton Client.

One example of such information is the path bandwidth. This can be captured in real time by end systems, in terms of transmission rate. On the other hand, the NSP can formulate preferences on given paths, at given time periods on given time scales.

One way for the NSP to provide guidance on highly dynamic network state information such as delay and load while preserving confidentiality and moderating processing load, is to provide them in a synthetic or abstract form, for example as a numerical indicator. It is important to have the possibility to reflect that the provided values are applicable for a given time period, for example busy hours or days, and are subject to changes over time. Also, how the NP has evaluated the cost associated to a given network state information is out of scope ot the ALTO protocol.

The usage of a time related cost is more proactive in that it can be used like a "time table" to figure out the best time to schedule data transfer and also anticipate predictable events including predictable flash crowds. The time-related information is not necessarily

historical and statistic. This is why the proposed time-sensitive Costs should be viewed as synthetic or as abstraction of real measurements rather than as statistics.

6.1.1. Path Occupation Cost and Endpointoccupationcost

The 'pathoccupationcost' can be used to reflect the NP view on the utilized path bandwidth at given time slots of given timeframes. For example, during daytime in the PC time zone, or between hh/mm and hh/mm in another time zone.

A Cost metric called 'endpointoccupationcost' and that accounts for the computation resources usage can be provided with the "dynamic" mode as well with a similar time scope updated for example with the knowledge of the NSP operating the CDN.

Both of these metrics are encoded with the 'JSONNumber' type.

Their optimal value is their minimal value.

They should also be provided in the numerical mode, that is as one single value, if requested so by the ALTO Client. This could be the case for an ALTO Server providing values at a short time scale (e.g. some minutes), or in the opposite, providing a timeless cost.

6.1.2. Dynamic Cost Attributes

For further extensions, specifications around "dynamic" costs are proposed and will be completed in further versions of this draft.

6.1.2.1. The dynamic Cost Mode

The "dynamic" mode applies to Costs that are eligible for the "numerical" Cost Mode and can also be expressed as such. In that sense, the "dynamic" mode is an extension of the "numerical" mode. Example are "pathoccupationcost" and "pathlosscost" that respectively report on the occupied bandwidth and packet loss on a path.

Other Cost Types such as JSONBool could also claim the "dynamic" mode, as states may be "true" or "false" depending on given periods. To stick to the target usage of Costs which is to be integrated in some evaluation process, an alternative could be to assign a numerical value to boolean costs and carefully map "true" and "false" with values '0' and '1'.

6.1.3. Proposed dynamic Cost Scope capability

To ensure that the Application Client uses the NP provided information on dynamic Costs in an inambuguous way, a capability called Cost Scope is introduced here, to report on the validity of the "dynamic" cost values. This draft focuses on the time related scope. Indeed, one may as well imagine some costs proposed in the future that could also be scoped w.r.t. space.

- Unit: ranging from "seconds" to "year", expresses the granularity of the information,
- Size: the scope of the dynamic cost, that is the number of units of the dynamic cost sample,
- Reference time zone,
- Begin: the index of the first unit in the sample,
- Next update: the date at which the sample will be re-computed,
- Last update: the last re-computation day.

Attributes 'Last update 'and 'Next update' report on the update frequency and age of the information.

6.1.3.1. Example of time scope for a dynamic cost

An example is: a metric called 'pathoccupationcost' (POC for short) is computed with a granularity of 2 hours, starting a 0h00, for 24 hours, with reference time GMT. The goal is to enable applications to see which time intervals in a day are the most favorable to operate.

6.1.4. Example of dynamic information resources in the IRD

The example IRD given in Section 4.1.1 includes an URI called "http://custom.alto.example.com/multi/endpointmulticost/lookup", in which the ALTO Server provides additional Mutliple Endpoint Costs including a Cost called "pathoccupationcost" for which the Cost Mode is "dynamic". This resource is accessible with other costs, via a separate subdomain called "custom.alto.example.com". The Endpoint Costs available via this subdomain are the "hopcount", "routingcost" and "pathoccupationcost" Cost Types, with the two first ones in the "numerical" Cost Mode and "pathoccupationcost" in the "dynamic" cost mode.

An ALTO Client can discover the services available by

"custom.alto.example.com" by successfully performing an OPTIONS request to "http://custom.alto.example.com/multi/endpointmulticost".

```
OPTIONS /multi/endpointmulticost HTTP/1.1
   Host: custom.alto.example.com
   Accept: application/alto-directory+json,application/alto-error+json


HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-directory+json

   {
     "resources" : [
      {
         "uri" : "http://custom.alto.example.com/multi/endpointmulticost",
         "media-types" : [ "application/alto-endpointmulticost+json" ],
         "accepts" : [ "application/alto-endpointmulticostparams+json" ],
         "capabilities" : {
           "cost-constraints" : true,
           "cost-modes" : [ "numerical", "numerical", "dynamic" ],
           "cost-types" : [ "routingcost", "hopcount", "pathoccupationcost" ],
           "cost-scope":  [ "permanent", "permanent",
                            {"unit": "hour", "size": 24, "begin": 0,
                             "time zone": "GMT",
                             "lastupdate": mm/hh/dd/mm/yyyy,
                             "nextupdate": mm/hh/dd/mm/yyyy}
           ]
         }
       }
     ]
   }
```

6.1.4.1. Example of response with a "dynamic" cost

To be completed

```
POST /endpointcost/lookup HTTP/1.1
  Host: alto.example.com
  Content-Length: [TODO]
  Content-Type: application/alto-endpointmulticostparams+json
  Accept: application/alto-endpointmulticost+json,application/alto-error+json

  {
    "cost-type" : ["routingcost", "pathoccupationcost"],
    "cost-mode" : ["numerical", "dynamic"],
    "endpoints" : {
      "srcs": [ "ipv4:192.0.2.2" ],
      "dsts": [
        "ipv4:192.0.2.89",
        "ipv4:198.51.100.34",
        "ipv4:203.0.113.45"
      ]
    }
  }


  HTTP/1.1 200 OK
  Content-Length: [TODO]
  Content-Type: application/alto-endpointmulticost+json

  {
    "meta" : {},
    "data" : {
      "cost-type" : ["routingcost", "pathoccupationcost"],
      "cost-mode" : ["numerical", "dynamic"],
      "map" : {
        "ipv4:192.0.2.2": {
          "ipv4:192.0.2.89"    : [1, [7, ..., 24 values]],
          "ipv4:198.51.100.34" : [2, [4, ..., 24 values]],
          "ipv4:203.0.113.45"  : [3, [2, ..., 24 values]]
        }
      }
    }
  }
```

7. IANA Considerations

Information for the ALTO Endpoint property registry maintained by the IANA and related to the new Endpoints supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Endpoint Property Registry" of

[ID-alto-protocol],

Information for the ALTO Cost Type Registry maintained by the IANA and related to the new Cost Types supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Cost Type Registry" of [ID-alto-protocol],

7.1. Information for IANA on proposed Cost Types

When a new ALTO Cost Type is defined, accepted by the ALTO working group and requests for IANA registration MUST include the following information, detailed in Section 11.2: Identifier, Intended Semantics, Security Considerations.

7.2. Information for IANA on proposed Endpoint Propeeries

Likewise, an ALTO Endpoint Property Registry could serve the same purposes as the ALTO Cost Type registry. Application to IANA registration for Endpoint Properties would follow a similar process.

8. Acknowledgements

Thank you to Dave Mac Dysan (Verizon) for fruitful discussions and comments on the last draft.

Sabine Randriamasy is partially supported by the MEDIEVAL project (http://www.ict-medieval.eu/), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the MEDIEVAL project or the European Commission.

Nico Schwan is partially supported by the ENVISION project (http://www.envision-project.org), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ENVISION project or the European Commission.

9. References

9.1. Normative References

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

[RFC5693] "Application Layer Traffic Optimization (ALTO) Problem Statement", October 2009.

9.2. Informative References

[ID-ALTO-Requirements7] "draft-ietf-alto-reqs-07.txt", January 2011.

[ID-alto-protocol] , Eds., ""ALTO Protocol" draft-ietf-alto-protocol-09.txt", June 2011.

[draft-jenkins-alto-cdn-use-cases-01] ""Use Cases for ALTO within CDNs" draft-jenkins-alto-cdn-use-cases-01", June 2011.

Authors' Addresses

Sabine Randriamasy (editor)
Alcatel-Lucent Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

Nico Schwan
Alcatel-Lucent Bell Labs
Lorenzstrasse 10
STUTTGART 70435
GERMANY

Email: Nico.Schwan@alcatel-lucent.com