

ALTO
Internet-Draft
Intended status: Standards Track
Expires: January 2, 2015

S. Kiesel
University of Stuttgart
R. Penno
Cisco Systems
July 1, 2014

Application-Layer Traffic Optimization (ALTO) Anycast Address
draft-kiesel-alto-ip-based-srv-disc-03

Abstract

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications that have to select one or several hosts from a set of candidates capable of providing a desired resource. ALTO is realized by a client-server protocol.

This document establishes a well-known IP address for the ALTO service and specifies how ALTO clients embedded in the resource consumer can use it to access the ALTO service.

Terminology and Requirements Language

This document makes use of the ALTO terminology defined in RFC 5693 [RFC5693].

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 2, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
2. ALTO Server Discovery based on well-known IP Address	5
2.1. ALTO Anycast IP Address (AAIPA)	5
2.2. ALTO Anycast Uniform Resource Identifier (AAURI)	5
2.3. ALTO Anycast Client Behavior	5
2.4. ALTO Anycast Server Behavior	6
3. Deployment Considerations	8
4. IANA Considerations	9
4.1. Registration of IPv4 Special Purpose Address	9
4.2. Registration of IPv6 Special Purpose Address	10
5. Security Considerations	12
6. References	13
6.1. Normative References	13
6.2. Informative References	13
Authors' Addresses	15

1. Introduction

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications that have to select one or several hosts from a set of candidates capable of providing a desired resource [RFC5693]. The ALTO requirements are itemized in [RFC6708]. The ALTO protocol [RFC7285] is a client-server protocol, which uses HTTP [RFC7230] for message transport.

Before an ALTO client can ask for guidance, it needs to discover one or more ALTO servers that can provide suitable guidance. Several procedures have been specified that produce a suitable HTTP URI for a given ALTO client (i.e., the URI may vary for different clients or different points of network attachment, etc.). These approaches are based on user input or DHCP [RFC7286], a "reverse DNS" (PTR) lookup [I-D.kist-alto-3pdisc], or redirection within the application protocol [I-D.kiesel-alto-alto4alto]. However, each of these approaches has technical or operational issues that will hinder the fast deployment of ALTO.

This document follows a different approach: it establishes a well-known address for the ALTO service to be used as application-layer anycast address. All ALTO clients seeking ALTO guidance are expected to send requests to this address. It is then the duty of "the network" to direct the query to a suitable server. This (re-)directing could be done on several layers, e.g., by resolving a well-known DNS domain name to different IP addresses (DNS split horizon), or by routing IP packets with the well-known IP address to different servers. This document follows the second option, as ALTO is closely related to IP routing and routing costs.

This document specifies a procedure that can be used if the ALTO client is embedded in the resource consumer. In other words, this document tries to meet requirement AR-32 in [RFC6708] while AR-33 is out of scope. Note that AR-20 mandates that "an ALTO client protocol must be designed in a way that the ALTO service can be provided by an entity that is not the operator of the underlying IP network." Though not violating said requirement, the procedure specified here is not helpful to fulfill it.

A more detailed discussion of various options where to place the functional entities comprising the overall ALTO architecture can be found in [I-D.ietf-alto-deployments].

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

2. ALTO Server Discovery based on well-known IP Address

2.1. ALTO Anycast IP Address (AAIPA)

IANA is requested to register (see Section 4) a single IPv4 address 192.0.0.X (TBD) and a single IPv6 address 2001:YYYY::ZZZZ (TBD) within the respective Special Purpose Address Registries as the well-known IP anycast addresses for the ALTO service. These addresses are called AAIPA (ALTO Anycast IP Address(es)) in this document.

2.2. ALTO Anycast Uniform Resource Identifier (AAURI)

The ALTO Anycast Uniform Resource Identifiers (AAURIs) are formed using the HTTP or HTTPS protocol identifier, the AAIPA in their literal forms (for literal IPv6 addresses in URIs see [RFC2732]), and a constant suffix. That is, there are four AAURIs (TBD: replace X, Y, Z with real values assigned by IANA):

`http://192.0.0.X/alto`

`https://192.0.0.X/alto`

`http://[2001:YYYY::ZZZZ]/alto`

`https://[2001:YYYY::ZZZZ]/alto`

2.3. ALTO Anycast Client Behavior

ALTO Clients that need to discover an ALTO server use the HTTP GET method [RFC7231] to access one AAURI, e.g.

`GET http://192.0.0.X/alto`

They MUST be prepared to receive an HTTP 307 temporary redirect to the ALTO server's Information Resource Directory URI (Sec. 9 of [RFC7285]).

For hosts equipped with multiple interfaces and/or using IPv4/v6 dual stack, this discovery method might yield different Information Resource Directory URIs for each interface and address family (i.e., IPv4/v6). In general, if a client wishes to communicate using one of its interfaces and using a specific IP address family, it SHOULD use this interface and the IP address associated with this interface to access the AAURI of the corresponding IP address family. Selecting an interface and IP address family, as well as comparing results returned from different ALTO servers, is out of the scope of this document.

TBD: rules for retrying (timers, etc.) in case of failure.

TBD: rules for caching discovery results.

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 re-run the ALTO server discovery procedure without relying on earlier gained information.

2.4. ALTO Anycast Server Behavior

ALTO anycast servers MUST listen on the IPv4 and/or IPv6 AAIPA(s) on the HTTPS ports for incoming HTTPS requests and they SHOULD listen on these AAIPA(S) on the HTTP port for incoming HTTP requests. They MUST answer GET requests to AAURI using the 307 (Temporary Redirect) status code and redirect to an ALTO server's Information Resource Directory URI.

The Information Resource Directory itself MUST NOT reside on a AAIPA, and it MUST NOT reside on an URI that resolves via DNS to a AAIPA. After issuing the 307 status code ALTO anycast servers MUST close the HTTP(S) connection.

Rationale for the requirements in the previous paragraph: The goal is to keep the TCP connection to the AAIPA as short as possible. When using anycast routing, IP packets belonging to an established TCP connection could be diverted to another ALTO anycast server due to state changes in the routing protocol or due to scheduled maintenance. Keeping the connection duration as short as possible reduces the risk of stalled or aborted connections. A UDP based lookup using one query packet and one reply packet (e.g., based on httpu) would eliminate that risk. However, there seems not to be a well-standardized candidate protocol and studies [Levine2006] suggest that short-lived TCP connections work well enough with anycast routing.

An ALTO anycast server MUST redirect an HTTPS request for an HTTPS AAURI to an HTTPS IRD URI. It MAY redirect an HTTP request for an HTTP AAURI to an HTTP IRD URI, but it MAY also redirect it to an HTTPS IRD URI.

The ALTO anycast server MAY consider the client's address and other information when generating the reply, in order to redirect to different ALTO servers depending on the client's identity or location within the network topology.

TBD: do we need some URI such as `http://192.0.0.X/server-identity` in

order to be able to identify the (misbehaving) ALTO anycast server that currently serves us?

TBD: how should the ALTO anycast server handle GET requests to other URIs or other HTTP methods?

3. Deployment Considerations

Network operators have to install one or more ALTO anycast servers as specified above. Depending on the the network deployment scenario they may use IP routing tables, HTTP proxies with URI rewriting, or other suitable mechanisms to direct GET-requests for a AAURI to one of these servers.

[TBD: explain in more detail] This works fine even with cascaded access routers with NATs. After each router hop the operator may decide whether to handle the discovery requests, e.g., using a static routing table entry, or whether let them flow "automatically" towards the internet backbones using the default routing table entry.

TBD: what happens if an operator does not deploy these scheme? Requests could be dropped at administrative borders. As an alternative, there could be "public" ALTO anycast servers to answer all queries that had not been answered in the respective originating access network. These servers could use the third-party ALTO server discovery procedure [I-D.kist-alto-3pdisc] to find the redirection target based on the client's IP address.

[TBD: explain in more detail] The advantage of this scheme is that it does not need support in home gateways, which would harm quick deployment. This scheme also doesn't need new interfaces between the operating system and applications, e.g., for passing DHCP options from the operating system to the application.

4. IANA Considerations

4.1. Registration of IPv4 Special Purpose Address

IANA is requested to register a single IPv4 address in the IANA IPv4 Special Purpose Address Registry [RFC5736].

[RFC5736] itemizes some information to be recorded for all designations:

1. The designated address prefix.

Prefix: TBD by IANA. Prefix length: /32

2. The RFC that called for the IANA address designation.

This document.

3. The date the designation was made.

TBD.

4. The date the use designation is to be terminated (if specified as a limited-use designation).

Unlimited. No termination date.

5. The nature of the purpose of the designated address (e.g., unicast experiment or protocol service anycast).

protocol service anycast.

6. For experimental unicast applications and otherwise as appropriate, the registry will also identify the entity and related contact details to whom the address designation has been made.

N/A.

7. The registry will also note, for each designation, the intended routing scope of the address, indicating whether the address is intended to be routable only in scoped, local, or private contexts, or whether the address prefix is intended to be routed globally.

Typically used within a network operator's network domain, but in principle globally routable.

8. The date in the IANA registry is the date of the IANA action, i.e., the day IANA records the allocation.

TBD.

4.2. Registration of IPv6 Special Purpose Address

IANA is requested to register a single IPv6 address in the IANA IPv6 Special Purpose Address Block [RFC4773].

[RFC4773] itemizes some information to be recorded for all designations:

1. The designated address prefix.

Prefix: TBD by IANA. Prefix length: /128

2. The RFC that called for the IANA address designation.

This document.

3. The date the designation was made.

TBD.

4. The date the use designation is to be terminated (if specified as a limited-use designation).

Unlimited. No termination date.

5. The nature of the purpose of the designated address (e.g., unicast experiment or protocol service anycast).

protocol service anycast.

6. For experimental unicast applications and otherwise as appropriate, the registry will also identify the entity and related contact details to whom the address designation has been made.

N/A.

7. The registry will also note, for each designation, the intended routing scope of the address, indicating whether the address is intended to be routable only in scoped, local, or private contexts, or whether the address prefix is intended to be routed globally.

Typically used within a network operator's network domain, but in principle globally routable.

8. The date in the IANA registry is the date of the IANA action, i.e., the day IANA records the allocation.

TBD.

5. Security Considerations

TBD

Issue: how to deal with TLS certificates for HTTPS?

TBD: rules for filtering route at administrative boundaries

6. References

6.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2732] Hinden, R., Carpenter, B., and L. Masinter, "Format for Literal IPv6 Addresses in URL's", RFC 2732, December 1999.
- [RFC4773] Huston, G., "Administration of the IANA Special Purpose IPv6 Address Block", RFC 4773, December 2006.
- [RFC5736] Huston, G., Cotton, M., and L. Vegoda, "IANA IPv4 Special Purpose Address Registry", RFC 5736, January 2010.
- [RFC7230] Fielding, R. and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, June 2014.
- [RFC7231] Fielding, R. and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, June 2014.

6.2. Informative References

- [I-D.ietf-alto-deployments]
Stiemerling, M., Kiesel, S., Previdi, S., and M. Scharf, "ALTO Deployment Considerations", draft-ietf-alto-deployments-09 (work in progress), February 2014.
- [I-D.kiesel-alto-alto4alto]
Kiesel, S., "Using ALTO for ALTO server selection", draft-kiesel-alto-alto4alto-00 (work in progress), July 2010.
- [I-D.kist-alto-3pdisc]
Kiesel, S., Krause, K., and M. Stiemerling, "Third-Party ALTO Server Discovery (3pdisc)", draft-kist-alto-3pdisc-05 (work in progress), January 2014.
- [Levine2006]
Levine, M., Lyon, B., and T. Underwood, "TCP Anycast - Don't believe the FUD. Operational experience with TCP and Anycast.", Presentation at NANOG37 <http://www.nanog.org/meetings/nanog37/presentations/matt.levine.pdf>, June 2006.
- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic

Optimization (ALTO) Problem Statement", RFC 5693, October 2009.

- [RFC6708] Kiesel, S., Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", RFC 6708, September 2012.
- [RFC7285] Alimi, R., Penno, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, June 2014.
- [RFC7286] Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M., and H. Song, "Application-Layer Traffic Optimization (ALTO) Server Discovery", RFC 7286, June 2014.

Authors' Addresses

Sebastian Kiesel
University of Stuttgart Information Center
Allmandring 30
Stuttgart 70550
Germany

Email: ietf-alto@skiesel.de
URI: <http://www.rus.uni-stuttgart.de/nks/>

Reinaldo Penno
Cisco Systems
170 West Tasman Dr
San Jose CA
USA

Email: repenno@cisco.com

ALTO
Internet-Draft
Intended status: Informational
Expires: January 5, 2015

S. Kiesel
University of Stuttgart
M. Stiemerling
H-DA
July 4, 2014

Application Layer Traffic Optimization (ALTO) Cross-Domain Server
Discovery
draft-kiesel-alto-xdom-disc-00

Abstract

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications that have to select one or several hosts from a set of candidates capable of providing a desired resource. ALTO is realized by a client-server protocol. Before an ALTO client can ask for guidance it needs to discover one or more ALTO servers that can provide suitable guidance.

In some deployment scenarios, in particular if the information about the network topology is partitioned and distributed over several ALTO servers, an ALTO client may need to discover an ALTO server outside of its own network domain, in order to get appropriate guidance.

This document details applicable scenarios, itemizes requirements, and analyzes existing solution approaches for such ALTO cross-domain server discovery. However, the specification of a procedure is beyond the scope of this document. Note, that in earlier versions of this document, ALTO cross-domain server discovery was referred to as "third-party discovery", but it has been renamed to avoid naming ambiguities.

Terminology and Requirements Language

This document makes use of the ALTO terminology defined in RFC 5693 [RFC5693].

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 5, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1. Document History	4
1.2. Feedback	4
2. The Need for ALTO Cross-Domain Server Discovery	5
2.1. Partitioned ALTO Knowledge	5
2.2. ALTO Queries on behalf of a Third Party	6
2.3. Partitioned Knowledge and Queries for a a Third Party	7
2.4. Example	7
3. Requirements for ALTO Cross-Domain Server Discovery	9
3.1. Discovery Client Application Programming Interface	9
3.2. Data Storage and Authority Requirements	9
3.3. Cross-Domain Operations Requirements	9
3.4. Protocol Requirements	10
3.5. Further Requirements	10
4. Related IETF Protocols and Activities	11
5. Security Considerations	12
6. IANA Considerations	13
7. References	14
7.1. Normative References	14
7.2. Informative References	14
Appendix A. ALTO and Tracker-based Peer-to-Peer Applications	16
Appendix B. Contributors List and Acknowledgments	21
Authors' Addresses	22

1. Introduction

Application-Layer Traffic Optimization (ALTO) [RFC5693] is realized by an HTTP-based client-server protocol [RFC7285]. Before an ALTO client can issue ALTO queries, it needs to discover a suitable ALTO server. During the design phase of the overall ALTO solution, two different server discovery scenarios have been identified and documented in the ALTO requirements document [RFC6708].

In the first scenario, documented in Req. AR-32, it is sufficient for a given ALTO client to discover a single ALTO server (or a small number of ALTO servers with identical knowledge, for reasons of reliability), usually the nearest one, as recommended by the operator of the access network. A discovery mechanism for this scenario, based on DHCP and DNS, is specified in [RFC7286]. An alternative approach, based on IP anycast, is documented in [I-D.kiesel-alto-ip-based-srv-disc].

In the second scenario, documented in Req. AR-33, an ALTO client may need to discover many different ALTO servers, depending on the queries it wants to issue. These ALTO servers may be located in other network domains than the client is. This document details applicable scenarios, itemizes requirements, and analyzes existing solution approaches for such ALTO cross-domain server discovery. However, the specification of a procedure is beyond the scope of this document. An experimental discovery procedure, which fulfills the requirements documented here, but which is currently not recommended for usage in the Internet, is documented in draft-kiesel-alto-xdom-disc-alg-00.txt.

1.1. Document History

This document is a direct successor of [I-D.kiesel-alto-3pdisc] and [I-D.kist-alto-3pdisc]. The scenario and mechanisms described here and in these documents have been referred to as "third-party server discovery" in the past. However, to avoid ambiguities with a completely different scenario, it has been renamed to "ALTO Cross-Domain Server Discovery".

1.2. Feedback

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

2. The Need for ALTO Cross-Domain Server Discovery

ALTO Cross-Domain Server discovery is needed, if two independent effects appear at the conjunction: partitioned knowledge and clients sending queries on behalf of other hosts. The following two subsections will discuss these effects separately, before the next subsection will discuss the combination of both.

2.1. Partitioned ALTO Knowledge

ALTO is realized by an HTTP-based client-server protocol. Basically, this protocol allows ALTO clients to download "cost maps", which indicate costs or other properties (according to various metrics) of the data path between endpoints. Furthermore, ALTO clients may access only parts of a cost map through the "map filtering service", and they may query endpoint properties through the "endpoint property service". Further similar services exist; for details refer to [RFC7285]. The endpoints are usually identified through their IP addresses. For efficiency reasons, endpoints may be grouped by using IP address prefixes and the "network map" with provider-defined identifiers (PIDs) defined within the ALTO protocol.

Conceptually, the ALTO protocol allows to query for path costs between arbitrary IP addresses from the whole IP address range, i.e., a full $2^{32} \times 2^{32}$ matrix for IPv4 and a $2^{128} \times 2^{128}$ matrix for IPv6 (The "PID" mechanism introduced in the ALTO protocol makes storage and transmission more efficient but does not alter that basic principle). And in fact, there may be deployment scenarios where a single ALTO server (or a cluster of servers operated by a single organization) has this "Internet-wide view", e.g., a community project collecting end-to-end measurements.

However, a very important class of scenarios is, when guiding information actually originates from the (access) network operators, such as Internet Service Providers (ISPs), IT departments of large companies or universities, etc. The information available at each of these providers will not be a full $N \times N$ matrix, but more like a $1 \times N$ vector, i.e., indicating cost "from us to anywhere" while cost "from anywhere to anywhere" is unknown. Several options exist how these pieces of information could be exposed to the ALTO clients, at least in theory:

1. The individual $1 \times N$ vectors could be aggregated to a virtual network-wide $N \times N$ matrix. This virtual matrix could be exposed through a single ALTO server or through a cluster of ALTO servers with identical information. However, no back-end protocol and process for this aggregation is currently defined.

2. Each (access) network operator could operate their own ALTO server, which only has this partial knowledge available there. Requests for unknown (source,destination)-pairs would be redirected to another ALTO server. The idea has been discussed [I-D.kiesel-alto-alto4alto] but no complete specification exists. A related option would be to establish something similar to a web search engine, which would do the redirecting. But again, no detailed specification exists.
3. Each (access) network operator could operate their own ALTO server, which only has this partial knowledge available there. It would be the duty of the client-side ALTO server discovery mechanism to directly find an ALTO server that can reasonably answer a given query. This option will be considered in more detail in the remainder of this memo.

2.2. ALTO Queries on behalf of a Third Party

An ALTO client may be co-located with or embedded into the resource consumer, i.e., the entity that seeks to access the desired resource and that will be one endpoint of the data transmission to be optimized. This kind of ALTO client will most often only be interested in "cost from me to somewhere else" queries, i.e., in an 1xN matrix.

In contrast, the ALTO client may also be located at some kind of directory server, P2P tracker, CDN redirect server, etc., i.e., at some entity that takes part in the application signaling but is not the endpoint of the actual user data transmission to be optimized. From an ALTO perspective, this kind of ALTO client does not issue ALTO queries for its own optimization needs, but instead it issues queries on behalf of remote third parties.

One motivation for this second type of configuration is faster deployment, as only some central servers would have to be equipped with an ALTO client and not all the clients. Furthermore, only these central servers (e.g., servers of a CDN) would need to access the ALTO servers, while the less-trusted clients could be denied to access the topology and cost maps. Another important reason is, that in some scenarios, much better optimization results can be achieved, if the ALTO guidance is considered at a central resource directory. See Appendix A for a detailed case study and analysis of such a scenario.

In the second scenario, ALTO queries may be interested in the path costs from an arbitrary point in the network topology (where the third party is, on behalf of which the query is sent) to other arbitrary points in the topology (where the candidate resource

providers are located). That is, such ALTO clients might want to access the full (virtual) NxN matrices.

A more detailed discussion of various options where to place the functional entities comprising the overall ALTO architecture can be found in [I-D.ietf-alto-deployments].

2.3. Partitioned Knowledge and Queries for a Third Party

As long as all ALTO servers store identical information, an ALTO client can send its queries to any server. A procedure for finding an ALTO server by means of DHCP is specified in [RFC7286].

If ALTO knowledge is partitioned and distributed over several ALTO servers and the ALTO client is co-located with a resource consumer, the DHCP based discovery procedure [RFC7286] will most likely work as well. The reason is, that this kind of ALTO client will only issue queries for "costs from me to anywhere", and the network operator can configure via DHCP an ALTO server that can answer these types of query.

If, however, ALTO knowledge is partitioned and distributed over several ALTO servers and the ALTO client issues ALTO queries on behalf of third parties, a different kind of server discovery mechanism is needed. These ALTO queries will ask for the "costs from X to anywhere" (where X is the IP address or PID of the third party). For each of these queries a suitable ALTO server has to be found and X will be the parameter for the discovery mechanism.

2.4. Example

The following, non-normative example illustrates the discovery procedure envisioned in this document.

Assume a peer-to-peer tracker is located in the network operated by ISP A. Some peer, which is located in ISP B's network, asks the tracker for an ALTO-optimized list of other peers that take part in a specific swarm. The tracker can observe the source address X of this message, which is the peer's IP address.

Assume that there is no omniscient ALTO server that knows the whole Internet topology. Therefore, the ALTO client in the tracker does a "back-connect" to the ALTO server operated by ISP B, which knows path costs from said peer (i.e., IP address X) to anywhere. The ALTO client retrieves this information and the tracker sorts the peer list according to it, before returning it to the peer.

Before this "back-connect" can occur, the server ALTO discovery

mechanism needs to map from the peer's IP address X to the ALTO server of ISP B, i.e., the network operator that controls IP address X and has assigned it to the peer.

3. Requirements for ALTO Cross-Domain Server Discovery

A solution for the problem described in the previous section would be an ALTO Cross-Domain Server Discovery system. This section itemizes requirements.

3.1. Discovery Client Application Programming Interface

The discovery client would be called through some kind of application programming interface (API) and the parameters would be an IP address and, for purposes of extensibility, a service identifier such as "ALTO". It would return an URI that offers the requested service ("ALTO") for the given IP address.

In other words, the client would be used to retrieve a mapping:

(IP address, "ALTO") -> IRD-URI

where IRD-URI is the URI of the Information Resource Directory (IRD, see Section 9 of [RFC7285]) of an ALTO server that can give reasonable guidance to a resource consumer with the indicated IP address.

3.2. Data Storage and Authority Requirements

The information for mapping IP addresses and service parameters to URIs should be stored in a - preferably distributed - database. It must be possible to delegate administration of parts of this database. Usually, the mapping from a specific IP address to an URI is defined by the authority that has administrative control over this IP address, e.g., the ISP in residential access networks or the IT department in enterprise, university, or similar networks.

3.3. Cross-Domain Operations Requirements

The cross-domain server discovery mechanism should be designed in such a way that it works across the public Internet and also in other IP-based networks. This in turn means that such mechanisms cannot rely on protocols that are not widely deployed across the Internet or protocols that require special handling within participating networks. An example is multicast, which is not generally available across the Internet.

The ALTO Cross-Domain Server Discovery protocol must support gradual deployment without a network-wide flag day. If the mechanism needs some kind of well-known "rendezvous point", re-using an existing infrastructure (such as the DNS root servers or the WHOIS database) should be preferred over establishing a new one.

3.4. Protocol Requirements

The protocol must be able to operate across middleboxes, especially across NATs and firewalls.

The protocol will specify an algorithm that determines the service parameters to be used when queries and responses are exchanged. This service parameter will specify 'ALTO' for the ALTO cross-domain service discovery. Potentially, it also specifies other required parameters needed for the service discovery, such as to be used transport or application level protocol.

The protocol will support the query with the above mentioned service parameters and allow that the response contains one or more URI(s).

The protocol shall not require any pre-knowledge from the client other than any information that is known to a regular IP host on the Internet.

3.5. Further Requirements

The ALTO cross domain server discovery cannot assume that the server discovery client and the server discovery responding entity are under the same administrative control.

4. Related IETF Protocols and Activities

TBD. Survey. In particular, the ECRIT WG has specified a reverse-DNS-based procedure [RFC7216] to solve a similar problem; TBD: analyze whether we can re-use or adapt it.

5. Security Considerations

A high-level discussion of security issues related to ALTO is part of the ALTO problem statement [RFC5693]. A classification of unwanted information disclosure risks, as well as specific security-related requirements can be found in the ALTO requirements document [RFC6708].

The remainder of this section focuses on security threats and protection mechanisms for the third-party ALTO server discovery procedure as such. Once the ALTO server's URI has been discovered and the communication between the ALTO client and the ALTO server starts, the security threats and protection mechanisms discussed in the ALTO protocol specification [RFC7285] apply.

TBD

6. IANA Considerations

This document does not require any IANA action.

7. References

7.1. Normative References

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

7.2. Informative References

- [I-D.ietf-alto-deployments]
Stiemerling, M., Kiesel, S., Previdi, S., and M. Scharf,
"ALTO Deployment Considerations",
draft-ietf-alto-deployments-09 (work in progress),
February 2014.
- [I-D.kiesel-alto-3pdisc]
Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M.,
Tomsu, M., and H. Song, "ALTO Server Discovery Protocol",
draft-kiesel-alto-3pdisc-05 (work in progress),
March 2011.
- [I-D.kiesel-alto-alto4alto]
Kiesel, S., "Using ALTO for ALTO server selection",
draft-kiesel-alto-alto4alto-00 (work in progress),
July 2010.
- [I-D.kiesel-alto-ip-based-srv-disc]
Kiesel, S. and R. Penno, "Application-Layer Traffic
Optimization (ALTO) Anycast Address",
draft-kiesel-alto-ip-based-srv-disc-03 (work in progress),
July 2014.
- [I-D.kist-alto-3pdisc]
Kiesel, S., Krause, K., and M. Stiemerling, "Third-Party
ALTO Server Discovery (3pdisc)", draft-kist-alto-3pdisc-05
(work in progress), January 2014.
- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic
Optimization (ALTO) Problem Statement", RFC 5693,
October 2009.
- [RFC6708] Kiesel, S., Previdi, S., Stiemerling, M., Woundy, R., and
Y. Yang, "Application-Layer Traffic Optimization (ALTO)
Requirements", RFC 6708, September 2012.
- [RFC7216] Thomson, M. and R. Bellis, "Location Information Server
(LIS) Discovery Using IP Addresses and Reverse DNS",
RFC 7216, April 2014.

- [RFC7285] Alimi, R., Penno, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, June 2014.
- [RFC7286] Kiesel, S., Stiernerling, M., Schwan, N., Scharf, M., and H. Song, "Application-Layer Traffic Optimization (ALTO) Server Discovery", RFC 7286, June 2014.

Appendix A. ALTO and Tracker-based Peer-to-Peer Applications

The ALTO protocol specification [RFC7285] details how an ALTO client can query an ALTO server for guiding information and receive the corresponding replies. However, in the considered scenario of a tracker-based P2P application, there are two fundamentally different possibilities where to place the ALTO client:

1. ALTO client in the resource consumer ("peer")
2. ALTO client in the resource directory ("tracker")

In the following, both scenarios are compared in order to explain the need for third-party ALTO queries.

In the first scenario (see Figure 2), the resource consumer queries the resource directory for the desired resource (F1). The resource directory returns a list of potential resource providers without considering ALTO (F2). It is then the duty of the resource consumer to invoke ALTO (F3/F4), in order to solicit guidance regarding this list.

In the second scenario (see Figure 4), the resource directory has an embedded ALTO client, which we will refer to as 3PAC (Third-Party ALTO Client) in this document. After receiving a query for a given resource (F1) the resource directory invokes the 3PAC to evaluate all resource providers it knows (F2/F3). Then it returns a, possibly shortened, list containing the "best" resource providers to the resource consumer (F4).

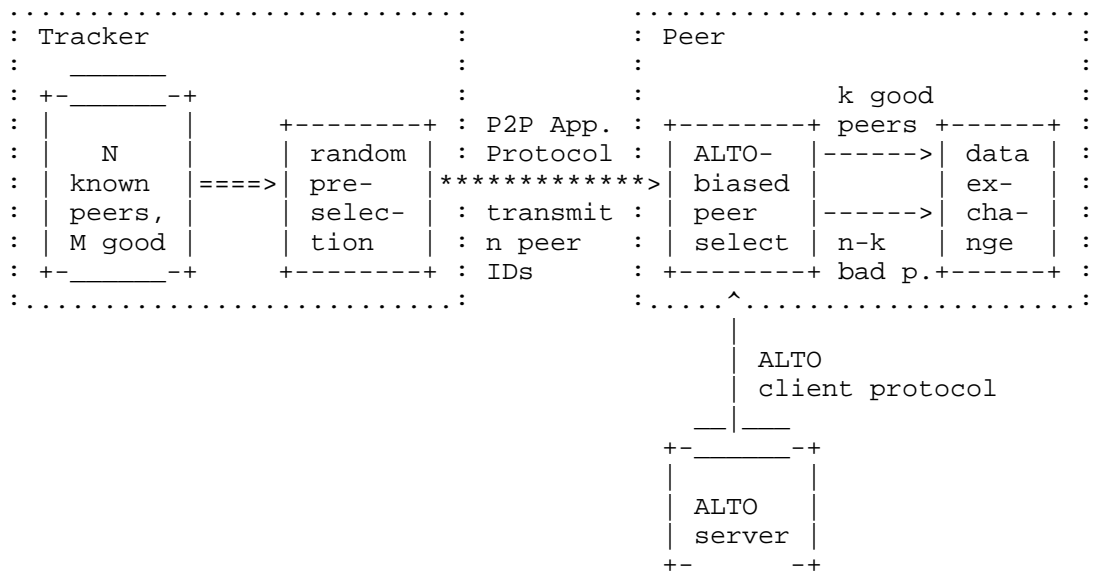


Figure 1: Tracker-based P2P Application with random peer preselection

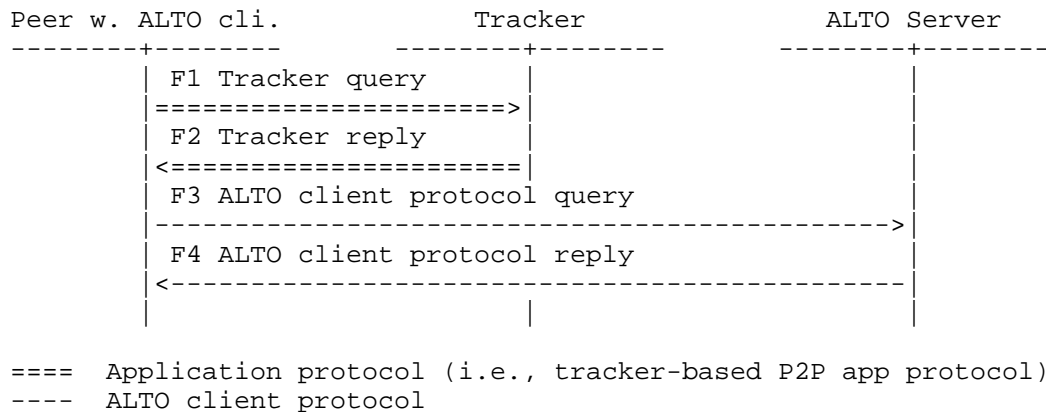


Figure 2: Basic message sequence chart for resource consumer-initiated ALTO query

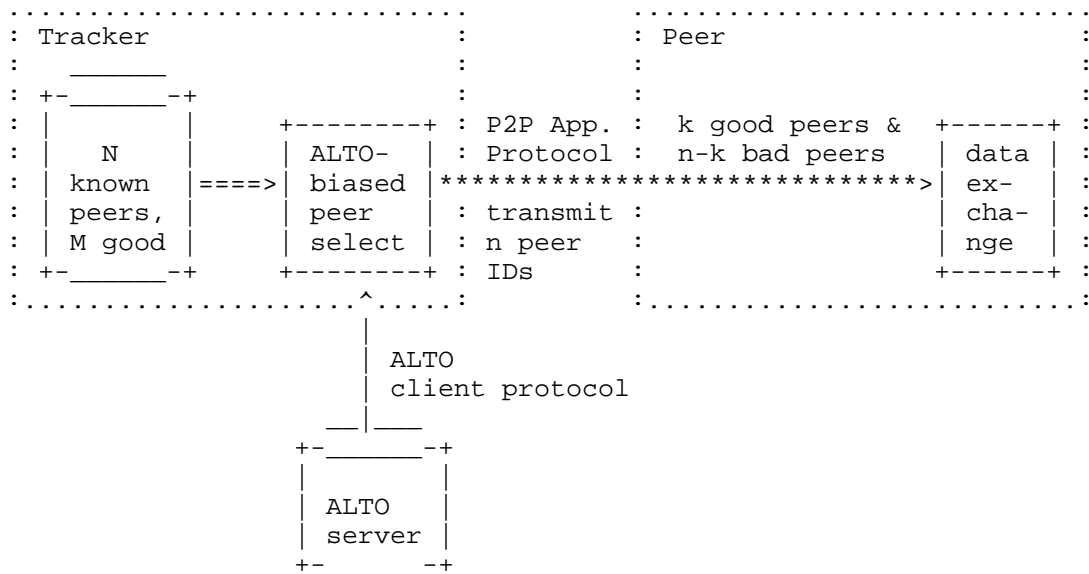


Figure 3: Tracker-based P2P Application with ALTO client in tracker

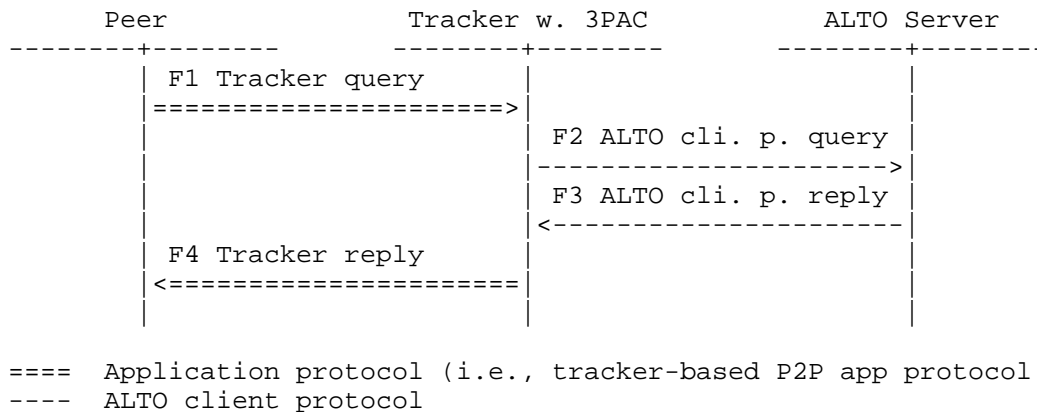


Figure 4: Basic message sequence chart for third-party ALTO query

Note: the message sequences depicted in Figure 2 and Figure 4 may occur both in the target-aware and the target-independent query mode (c.f. [RFC6708]). In the target-independent query mode no message exchange with the ALTO server might be needed after the tracker query, because the candidate resource providers could be evaluated using a locally cached "map", which has been retrieved from the ALTO server some time ago.

The problem with the first approach is, that while the resource directory might know thousands of peers taking part in a swarm, the list returned to the resource consumer is usually shortened for efficiency reasons. Therefore, the "best" (in the sense of ALTO) potential resource providers might not be contained in that list anymore, even before ALTO can consider them.

For illustration, consider a simple model of a swarm, in which all peers fall into one of only two categories: assume that there are "good" ("good" in the sense of ALTO's better-than-random peer selection, based on an arbitrary desired rating criterion) and "bad" peers only. Having more different categories makes the maths more complex but does not change anything to the basic outcome of this analysis. Assume that the swarm has a total number of N peers, out of which are M "good" and $N-M$ "bad" peers, which are all known to the tracker. A new peer wants to join the swarm and therefore asks the tracker for a list of peers.

If, according to the first approach, the tracker randomly picks n peers from the N known peers, the result can be described with the hypergeometric distribution. The probability that the tracker reply contains exactly k "good" peers (and $n-k$ "bad" peers) is:

$$P(X=k) = \frac{\frac{\binom{m}{k} \binom{N-m}{n-k}}{\binom{N}{n}}}{\frac{\binom{N}{n}}{\binom{N}{n}}}$$

$$\text{with } \frac{\binom{n}{k}}{\binom{n}{k}} = \frac{n!}{k! (n-k)!} \quad \text{and} \quad n! = n * (n-1) * (n-2) * \dots * 1$$

The probability that the reply contains at most k "good" peers is:
 $P(X \leq k) = P(X=0) + P(X=1) + \dots + P(X=k)$.

For example, consider a swarm with $N=10,000$ peers known to the tracker, out of which $M=100$ are "good" peers. If the tracker randomly selects $n=100$ peers, the formula yields for the reply: $P(X=0)=36\%$, $P(X \leq 4)=99\%$. That is, with a probability of approx. 36% this list does not contain a single "good" peer, and with 99% probability there are only four or less of the "good" peers on the list. Processing this list with the guiding ALTO information will ensure that the few favorable peers are ranked to the top of the

list; however, the benefit is rather limited as the number of favorable peers in the list is just too small.

Much better traffic optimization could be achieved if the tracker would evaluate all known peers using ALTO, and return a list of 100 peers afterwards. This list would then include a significantly higher fraction of "good" peers. (Note, that if the tracker returned "good" peers only, there might be a risk that the swarm might disconnect and split into several disjunct partitions. However, finding the right mix of ALTO-biased and random peer selection is out of the scope of this document.)

Therefore, from an overall optimization perspective, the second scenario with the ALTO client embedded in the resource directory is advantageous, because it is ensured that the addresses of the "best" resource providers are actually delivered to the resource consumer. An architectural implication of this insight is that the ALTO server discovery procedures must support third-party discovery. That is, as the tracker issues ALTO queries on behalf of the peer which contacted the tracker, the tracker must be able to discover an ALTO server that can give guidance suitable for that respective peer.

Appendix B. Contributors List and Acknowledgments

The initial version of this document was co-authored by Marco Tomsu (Alcatel-Lucent).

This document borrows some text from [RFC7286], as it was historically part of that memo. Special thanks to Michael Scharf and Nico Schwan.

Authors' Addresses

Sebastian Kiesel
University of Stuttgart Information Center
Allmandring 30
Stuttgart 70550
Germany

Email: ietf-alto@skiesel.de
URI: <http://www.rus.uni-stuttgart.de/nks/>

Martin Stiernerling
University of Applied Sciences Darmstadt, Computer Science Dept.
Haardtring 100
Darmstadt 64295
Germany

Phone: +49 6151 16 7938
Email: mls.ietf@gmail.com
URI: <http://ietf.stiernerling.org>

ALTO
Internet-Draft
Intended status: Experimental
Expires: January 5, 2015

S. Kiesel
K. Krause
University of Stuttgart
M. Stiemerling
H-DA
July 4, 2014

Application Layer Traffic Optimization (ALTO) Cross-Domain Server
Discovery - Experimental Algorithm Specification
draft-kiesel-alto-xdom-disc-alg-00

Abstract

The goal of Application-Layer Traffic Optimization (ALTO) is to provide guidance to applications that have to select one or several hosts from a set of candidates capable of providing a desired resource. ALTO is realized by a client-server protocol. Before an ALTO client can ask for guidance it needs to discover one or more ALTO servers that can provide suitable guidance.

This document contains a strawman proposal for an ALTO Cross-Domain Server Discovery procedure (also known as Third-Party Discovery). Technically, the algorithm specified in this document takes one IP address and a U-NAPTR Service Parameter (i.e., "ALTO:http" or "ALTO:https") as parameters. It performs several DNS lookups (for U-NAPTR and SOA resource records) and returns one or more URI(s) of information resources related to that IP address.

The functionality has been validated in a lab environment. However, the feasibility and possible side-effects of Internet-wide "production use" are not yet understood. The purpose of this document is to foster further discussion within the ALTO working group. Readers of this document should exercise caution in evaluating its value for implementation and deployment.

Terminology and Requirements Language

This document makes use of the ALTO terminology defined in RFC 5693 [RFC5693].

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 5, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1. Document History	4
1.2. Feedback	4
2. ALTO Cross-Domain Server Discovery Procedure Specification . .	5
2.1. Interface	5
2.2. Basic Principle	5
2.3. Overall Procedure	6
2.4. Specification of Tasks and Conditional Branches	7
2.4.1. T1: Prepare Domain Name for Reverse DNS Lookup	7
2.4.2. T2/B1: U-NAPTR Lookup in Reverse Zone	7
2.4.3. B2/T3/B3: Acquire SOA Record for Reverse Zone	8
2.4.4. T4/B4: U-NAPTR Lookup on SOA-MNAME	9
3. Implementation, Deployment, and Operational Considerations . .	10
3.1. Considerations for ALTO Clients	10
3.1.1. Resource Consumer Initiated Discovery	10
3.1.2. IPv4/v6 Dual Stack, Multihoming, NAT, and Host Mobility	10
3.2. Deployment Considerations for Network Operators	11
3.2.1. NAPTR in Reverse Tree vs. SOA-based discovery	11
3.2.2. Separation of Interests	11
3.3. Impact on DNS	12
3.3.1. Non-PTR Resource Records in Reverse Tree	12
3.3.2. Usage with DNS Hidden Master Servers	12
3.3.3. Load on the DNS	12
4. Security Considerations	13
4.1. Integrity of the ALTO Server's URI	13
4.2. Availability of the ALTO Server Discovery Procedure	14
4.3. Confidentiality of the ALTO Server's URI	14
4.4. Privacy for ALTO Clients	15
5. IANA Considerations	16
6. References	17
6.1. Normative References	17
6.2. Informative References	17
Authors' Addresses	19

1. Introduction

The motivation for Application-Layer Traffic Optimization (ALTO) Cross-Domain Server Discovery (also known as Third-Party Server Discovery), as well as applicable scenarios and requirements for a solution are documented in draft-kiesel-alto-xdom-disc-00. In the following, we assume that the reader is familiar with said document.

This document presents the specification of an DNS-based procedure that satisfies these requirements. The functionality has been validated in a lab environment. However, the feasibility and possible side-effects of Internet-wide "production use" are not yet understood.

The purpose of this document is to foster further discussion within the ALTO working group and the IETF community in general. Readers of this document should exercise caution in evaluating its value for implementation and deployment.

1.1. Document History

This document is a direct successor of [I-D.kiesel-alto-3pdisc] and [I-D.kist-alto-3pdisc]. The scenario and mechanisms described here and in these documents have been referred to as "third-party server discovery" in the past. However, to avoid ambiguities with a completely different scenario, it has been renamed to "ALTO Cross-Domain Server Discovery".

1.2. Feedback

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

2. ALTO Cross-Domain Server Discovery Procedure Specification

2.1. Interface

The algorithm specified in this document takes one IP address and a U-NAPTR Service Parameter (i.e., "ALTO:http" or "ALTO:https") as parameters. It performs several DNS lookups (for U-NAPTR and SOA resource records) and returns one or more URI(s) of information resources related to that IP address.

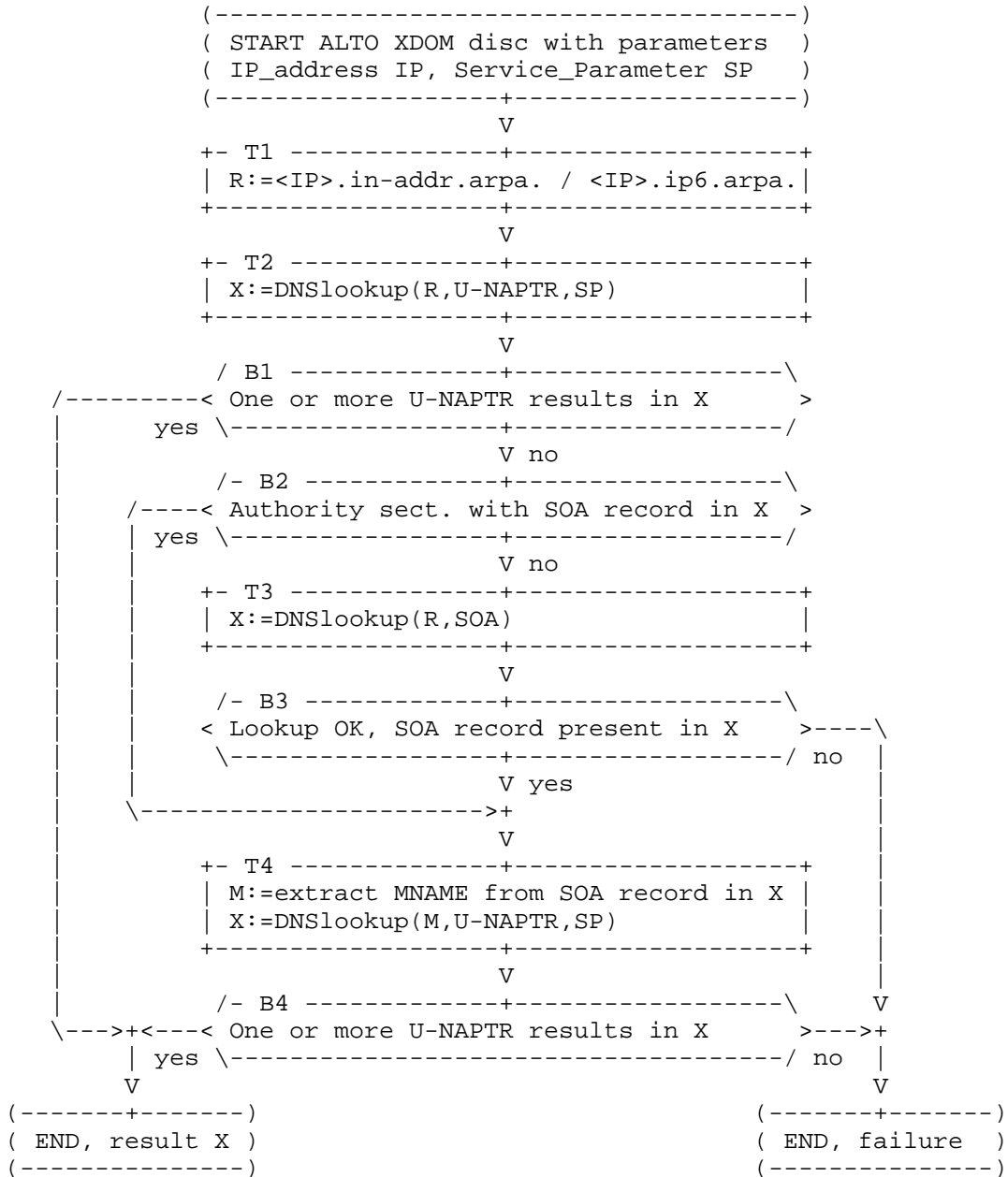
2.2. Basic Principle

The algorithm sequentially tries two different lookup strategies. First, an ALTO-specific U-NAPTR lookup is performed in the "reverse tree", i.e., in subdomains of in-addr.arpa. or ip6.arpa., respectively. If this lookup does not yield a usable result, the SOA record for the reverse zone is acquired, its master name server (MNAME) value is extracted and used for a further ALTO-specific U-NAPTR lookup.

The goal is to allow deployment scenarios that require fine-grained discovery on a per-IP basis, as well as large-scale scenarios where discovery is to be enabled for a large number of IP addresses with a small number of additional DNS resource records.

2.3. Overall Procedure

This figure gives an overview on the discovery procedure. All tasks (T) and conditional branches (B) are specified below.



2.4.1. T1: Prepare Domain Name for Reverse DNS Lookup

`https://altoserver.isp.example.net/secure/directory` or the HTTP URI `http://altoserver.isp.example.net/directory`, with the former being preferred.

`3.100.51.198.in-addr.arpa.`

```
IN NAPTR 100 10 "u" "ALTO:https"
"!.*!https://altoserver.isp.example.net/secure/directory!" ""
```

```
IN NAPTR 200 10 "u" "ALTO:http"
"!.*!http://altoserver.isp.example.net/directory!" ""
```

Conditional Branch B1 checks whether at least one U-NAPTR record matching the service parameter SP could be retrieved. If so, the procedure ends successfully and the sorted list of U-NAPTR records is the result. Otherwise, if no U-NAPTR records could be retrieved, we continue with B2.

Note: The U-NAPTR lookup in Task T2 is identical to Step 2 specified in [RFC7286], which specifies with "manual input" and "DHCP" two alternatives for acquiring the name to be looked up. Therefore, it is possible to merge both documents into a common ALTO server discovery framework.

2.4.3. B2/T3/B3: Acquire SOA Record for Reverse Zone

The task of B2/T3/B3 is to acquire the SOA record for the "reverse zone", i.e., the zone in the `in-addr.arpa.` or `ip6.arpa.` domain that contains the IP address in question.

A sample SOA record could be:

```
100.51.198.in-addr.arpa
IN SOA dns1.isp.example.net. hostmaster.isp.example.net. (
                                1           ; Serial
                                604800      ; Refresh
                                86400       ; Retry
                                2419200     ; Expire
                                604800 )    ; Negative Cache TTL
```

Conditional Branch B2 checks whether the SOA record was present in the authority section of X, i.e., the result of Task T2. If not, an explicit lookup is done in Task T3. If Conditional Branch B3 determines that this explicit lookup failed, the discovery procedure is aborted without a result; otherwise we continue with T4.

2.4.4. T4/B4: U-NAPTR Lookup on SOA-MNAME

Now that the SOA record is available, Task T4 first extracts the MNAME field, i.e., the responsible master name server from the SOA record. An example MNAME could be:

dns1.isp.example.net.

Then, a U-NAPTR lookup as specified in Task T2 is performed on this MNAME and the result is stored in variable "X".

Conditional Branch B4 checks whether at least one U-NAPTR record matching the service parameter SP could be retrieved. If so, the procedure ends successfully and the sorted list of U-NAPTR records is the result. Otherwise, if no U-NAPTR records could be retrieved, the discovery procedure is aborted without a result.

3. Implementation, Deployment, and Operational Considerations

3.1. Considerations for ALTO Clients

3.1.1. Resource Consumer Initiated Discovery

To some extent, ALTO requirement AR-32 [RFC6708], i.e., resource consumer initiated ALTO server discovery, can be seen as a special case of cross-domain ALTO server discovery. To that end, an ALTO client embedded in a resource consumer would have to figure out its own "public" IP address and perform the procedures described in this document on that address. However, due to the widespread deployment of Network Address Translators (NAT), additional protocols and mechanisms such as STUN [RFC5389] would be needed and considerations for UNSAF [RFC3424] apply. Therefore, using the procedures specified in this document for resource consumer based ALTO server discovery is generally NOT RECOMMENDED. Note that a less versatile yet simpler approach for resource consumer initiated ALTO server discovery is specified in [RFC7286].

3.1.2. IPv4/v6 Dual Stack, Multihoming, NAT, and Host Mobility

The algorithm specified in this document can discover ALTO server URIs for a given IP address. The intention is, that an entity taking part in the application signaling (e.g., a resource directory such as a P2P tracker) that receives signaling messages from a resource consumer can use the source address contained in these messages to discover suitable ALTO servers for this specific resource consumer.

However, resource consumers (as defined in Section 2 of [RFC5693]) may reside on hosts with more than one IP address, e.g., due to IPv4/v6 dual stack operation and/or multihoming. IP packets sent with different source addresses may be subject to different routing policies and path costs. In some deployment scenarios, it may even be required to ask different sets of ALTO servers for guidance. Furthermore, source addresses in IP packets may be modified en-route by Network Address Translators (NAT).

If a resource consumer queries a resource directory for candidate resource providers, the locally selected (and possibly en-route translated) source address of the query message - as observed by the resource directory - will become the basis for the ALTO server discovery and the subsequent optimization of the resource directory's reply. If, however, the resource consumer then selects different source addresses to contact returned resource providers, the desired better-than-random "ALTO effect" may not occur.

Therefore, a dual stack or multihomed resource consumer SHOULD either

always use the same address for contacting the resource directory and the resource providers, i.e., overriding the operating system's automatic source IP address selection, or use resource consumer based ALTO server discovery [RFC7286] to discover suitable ALTO servers for every local address and then locally perform ALTO-influenced resource consumer selection and source address selection. Similarly, resource consumers on mobile hosts SHOULD query the resource directory again after a change of IP address, in order to get a list of candidate resource providers that is optimized for the new IP address.

3.2. Deployment Considerations for Network Operators

3.2.1. NAPTR in Reverse Tree vs. SOA-based discovery

As already outlined in Section 2.2, the ALTO cross-domain server discovery procedure sequentially tries two different lookup strategies, thus giving network operators the choice of two different deployment options:

- o Individual NAPTR records in the in-addr.arpa or ip6.arpa domains allow very fine-grained discovery of ALTO "entry point" URIs on a per-IP-address basis. This method also gives the fastest response times and causes a comparatively low load on the DNS, as the algorithm terminates successfully after the first DNS query. DNS operators that already maintain reverse zones (e.g., for PTR records) should prefer this option, possibly using DNS server implementation-specific methods for mass deployment (e.g., BIND9's \$GENERATE statement).
- o If a DNS operator considers the first option too cumbersome, or if IPv6 privacy extensions is to be used without dynamic PTR updates, setting up SOA records in the in-addr.arpa. or ip6.arpa. subdomains plus setting up corresponding ALTO-specific U-NAPTR records will also give reasonable, yet less fine-grained results at the cost of slightly higher delay and load on the DNS.

3.2.2. Separation of Interests

We assume that if two organizations share parts of their DNS infrastructure, i.e., have a common SOA record in their in-addr.arpa. or ip6.arpa. subdomain(s), they will also be able to operate a common ALTO server, which still may do redirections if desired or required by policies.

Note that the ALTO server discovery procedure is supposed to produce only a first URI of an ALTO server that can give reasonable guidance to the client. An ALTO server can still return different results based on the client's address (or other identifying properties) or

redirect the client to another ALTO server using mechanisms of the ALTO protocol (see Sect. 9 of [RFC7285]).

3.3. Impact on DNS

3.3.1. Non-PTR Resource Records in Reverse Tree

Installing NAPTR records, i.e., a record type other than PTR records, in the in-addr.arpa or ip6.arpa domain may seem uncommon, but it is not a new concept. Earlier documents that specify the usage of Non-PTR resource records in the reverse tree include RFC 4025 [RFC4025], RFC 4255 [RFC4255], and RFC 4322 [RFC4322].

3.3.2. Usage with DNS Hidden Master Servers

In some deployment scenarios, the Master DNS server for a in-addr.arpa. or ip6.arpa. subdomain, as indicated in the respective SOA record, may not be reachable due to traffic restrictions ("hidden master"). This does not cause any problems with the algorithm described here, as the MNAME is only used for further DNS lookups; but it is never attempted to contact this server directly.

3.3.3. Load on the DNS

The procedure described in this document features several nested conditional branches, but no loops. Each time being called it attempts one to three DNS lookups.

4. Security Considerations

A high-level discussion of security issues related to ALTO is part of the ALTO problem statement [RFC5693]. A classification of unwanted information disclosure risks, as well as specific security-related requirements can be found in the ALTO requirements document [RFC6708].

The remainder of this section focuses on security threats and protection mechanisms for the ALTO cross-domain server discovery procedure as such. Once the ALTO server's URI has been discovered and the communication between the ALTO client and the ALTO server starts, the security threats and protection mechanisms discussed in the ALTO protocol specification [RFC7285] apply.

4.1. Integrity of the ALTO Server's URI

Scenario Description

An attacker could compromise the ALTO server discovery procedure or infrastructure in a way that ALTO clients would discover a "wrong" ALTO server URI.

Threat Discussion

This is probably the most serious security concern related to ALTO server discovery. The discovered "wrong" ALTO server might not be able to give guidance to a given ALTO client at all, or it might give suboptimal or forged information. In the latter case, an attacker could try to use ALTO to affect the traffic distribution in the network or the performance of applications (see also Section 15.1. of [RFC7285]). Furthermore, a hostile ALTO server could threaten user privacy (see also Section 5.2.1, case (5a) in [RFC6708]).

However, it should also be noted that, if an attacker was able to compromise the DNS infrastructure used for ALTO cross-domain server discovery (see below), (s)he could also launch significantly more serious other attacks (e.g., redirecting various application protocols).

Protection Strategies and Mechanisms

The ALTO cross-domain server discovery procedure relies on a series of DNS lookups. If an attacker was able to modify or spoof any of the DNS records, the resulting URI could be replaced by a forged URI. The application of DNS security (DNSSEC) [RFC4033] provides a means to limit attacks that rely on modification of the DNS records while in transit. Additional operational precautions for safely operating the DNS infrastructure are required in order to ensure that name servers do not sign forged (or otherwise

"wrong") resource records. Security considerations specific to U-NAPTR are described in more detail in [RFC4848].

A related risk is the impersonation of the ALTO server (i.e., attacks after the correct URI has been discovered). This threat and protection strategies are discussed in Section 15.1 of [RFC7285]. Note that if TLS is used to protect ALTO, the server certificate will contain the host name (CN). Consequently, only the host part of the HTTPS URI will be authenticated, i.e., the result of the ALTO server discovery procedure. The DNS/U-NAPTR based mapping within the ALTO cross-domain server discovery procedure needs to be secured as described above, e.g., by using DNSSEC.

In addition to active protection mechanisms, users and network operators can monitor application performance and network traffic patterns for poor performance or abnormalities. If it turns out that relying on the guidance of a specific ALTO server does not result in better-than-random results, the usage of the ALTO server may be discontinued (see also Section 15.2 of [RFC7285]).

4.2. Availability of the ALTO Server Discovery Procedure

Scenario Description

An attacker could compromise the ALTO cross-domain server discovery procedure or infrastructure in a way that ALTO clients would not be able to discover any ALTO server.

Threat Discussion

If no ALTO server can be discovered (although a suitable one exists) applications have to make their decisions without ALTO guidance. As ALTO could be temporarily unavailable for many reasons, applications must be prepared to do so. However, The resulting application performance and traffic distribution will correspond to a deployment scenario without ALTO.

Protection Strategies and Mechanisms

Operators should follow best current practices to secure their DNS and ALTO (see Section 15.5 of [RFC7285]) servers against Denial-of-Service (DoS) attacks.

4.3. Confidentiality of the ALTO Server's URI

Scenario Description

An unauthorized party could invoke the ALTO cross-domain server discovery procedure, or intercept discovery messages between an authorized ALTO client and the DNS servers, in order to acquire knowledge of the ALTO server URI for a specific resource consumer.

Threat Discussion

In the ALTO use cases that have been described in the ALTO problem statement [RFC5693] and/or discussed in the ALTO working group, the ALTO server's URI as such has always been considered as public information that does not need protection of confidentiality.

Protection Strategies and Mechanisms

No protection mechanisms for this scenario have been provided, as it has not been identified as a relevant threat. However, if a new use case is identified that requires this kind of protection, the suitability of this ALTO server discovery procedure as well as possible security extensions have to be re-evaluated thoroughly.

4.4. Privacy for ALTO Clients

Scenario Description

An unauthorized party could intercept messages between an ALTO client and the DNS servers, and thereby find out the fact that said ALTO client uses (or at least tries to use) the ALTO service on behalf of a specific resource consumer.

Threat Discussion

In the ALTO use cases that have been described in the ALTO problem statement [RFC5693] and/or discussed in the ALTO working group, this scenario has not been identified as a relevant threat.

Protection Strategies and Mechanisms

No protection mechanisms for this scenario have been provided, as it has not been identified as a relevant threat. However, if a new use case is identified that requires this kind of protection, the suitability of this ALTO server discovery procedure as well as possible security extensions have to be re-evaluated thoroughly.

5. IANA Considerations

This document does not require any IANA action.

This document specifies an algorithm that uses U-NAPTR lookups [RFC4848] with the Application Service Tag "ALTO" and the Application Protocol Tags "http" and "https". These tags have already been registered with IANA. In particular, for the registration of the Application Service Tag "ALTO", see [RFC7286].

6. References

6.1. Normative References

- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3403] Mealling, M., "Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database", RFC 3403, October 2002.
- [RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, October 2003.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [RFC4848] Daigle, L., "Domain-Based Application Service Location Using URIs and the Dynamic Delegation Discovery Service (DDDS)", RFC 4848, April 2007.

6.2. Informative References

- [I-D.kiesel-alto-3pdisc] Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M., Tomsu, M., and H. Song, "ALTO Server Discovery Protocol", draft-kiesel-alto-3pdisc-05 (work in progress), March 2011.
- [I-D.kist-alto-3pdisc] Kiesel, S., Krause, K., and M. Stiemerling, "Third-Party ALTO Server Discovery (3pdisc)", draft-kist-alto-3pdisc-05 (work in progress), January 2014.
- [RFC3424] Daigle, L. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.
- [RFC4025] Richardson, M., "A Method for Storing IPsec Keying Material in DNS", RFC 4025, March 2005.
- [RFC4255] Schlyter, J. and W. Griffin, "Using DNS to Securely Publish Secure Shell (SSH) Key Fingerprints", RFC 4255,

January 2006.

- [RFC4322] Richardson, M. and D. Redelmeier, "Opportunistic Encryption using the Internet Key Exchange (IKE)", RFC 4322, December 2005.
- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.
- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.
- [RFC6708] Kiesel, S., Previdi, S., Stiernerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", RFC 6708, September 2012.
- [RFC7285] Alimi, R., Penno, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Protocol", RFC 7285, June 2014.
- [RFC7286] Kiesel, S., Stiernerling, M., Schwan, N., Scharf, M., and H. Song, "Application-Layer Traffic Optimization (ALTO) Server Discovery", RFC 7286, June 2014.

Authors' Addresses

Sebastian Kiesel
University of Stuttgart Information Center
Allmandring 30
Stuttgart 70550
Germany

Email: ietf-alto@skiesel.de
URI: <http://www.rus.uni-stuttgart.de/nks/>

Kilian Krause
University of Stuttgart Information Center
Allmandring 30
Stuttgart 70550
Germany

Email: schreibt@normalerweise.net
URI: <http://www.rus.uni-stuttgart.de/nks/>

Martin Stiernerling
University of Applied Sciences Darmstadt, Computer Science Dept.
Haardtring 100
Darmstadt 64295
Germany

Phone: +49 6151 16 7938
Email: mls.ietf@gmail.com
URI: <http://ietf.stiernerling.org>

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 5, 2015

S. Randriamasy
W. Roome
Alcatel-Lucent
N. Schwan
Thales Deutschland
July 4, 2014

Multi-Cost ALTO
draft-randriamasy-alto-multi-cost-08

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 simple way to extend the information provided by the current ALTO protocol in two ways. First, including information on multiple Cost Types in a single ALTO transaction provides a better mapping of the Selected Endpoints to needs of the growing diversity of Content and Resources Networking Applications and to the network conditions. Second, one ALTO query and response exchange on N Cost Types is faster and more efficient than N single cost transactions. All this also helps producing a faster and more robust choice when multiple Endpoints need to be selected. Last, the draft proposes to enrich the filtering capabilities by allowing constraints involving several metrics combined by several types of logical operators. This allows the applications to set finer requirements and above all to include compromises on those requirements.

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 5, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
2. Application Scope And Terminology	5
3. Uses Cases For Using Multiple Costs	6
3.1. Use Cases For Using Additional Costs	6
3.1.1. Delay Sensitive Overlay Applications	7
3.1.2. Selection Of Physical Servers Involved In Virtualized Applications	7
3.1.3. CDN Surrogate Selection	8
3.1.4. Some Proposed Additional Properties And Costs	9
3.2. Use Cases For Multi-Cost ALTO Transactions	10
3.2.1. Optimized Endpoint Cost Service	10
3.2.2. Optimized Filtered Cost Map Service	11
3.2.3. Cases Of Unpredictable Endpoint Cost Value Changes	11
3.2.3.1. Case Of A Multi-Cost ALTO Query Upon A Route Change	12
3.2.3.2. Case Of A Multi-Cost ALTO Query Upon A Cost Value Change	13
4. ALTO Protocol Updates Needed To Support Multi-Cost Transactions	14
4.1. List Of ALTO Protocol Updates Required And Recommended	15
4.2. Updates Required In The Member Format Of Objects	15

4.2.1.	Cost Value Encoded In JSONArray	16
4.2.2.	Scalar 'cost-type' Member Replaced By Array 'cost- types' Member	16
4.2.3.	Rule On Cost Value Order In ALTO Responses	17
4.3.	Updates Recommended In The Object Structure	17
5.	Extended Constraints On Multi-Cost Values	17
5.1.	Use Cases For Multi-Cost Multi-Operator Constraints	18
5.2.	Extended constraints in Multi-Cost ALTO	18
6.	Protocol Extensions For Multi-Cost ALTO Transactions	19
6.1.	Information Resources Directory	19
6.1.1.	Example of Multi-Cost specific resources in the IRD	20
6.2.	Multi-Cost Map Service	22
6.2.1.	Media Type	22
6.2.2.	HTTP Method	22
6.2.3.	Input Parameters	22
6.2.4.	Capabilities	22
6.2.5.	Uses	23
6.2.6.	Response	23
6.2.7.	Example	24
6.3.	Filtered Multi-Cost Map	24
6.3.1.	Media Type	25
6.3.2.	HTTP Method	25
6.3.3.	Input Parameters	25
6.3.4.	Capabilities	27
6.3.5.	Uses	27
6.3.6.	Response	27
6.3.7.	Example 1	28
6.3.8.	Example 2	28
6.4.	Endpoint Multi-Cost Service	29
6.4.1.	Media Type	30
6.4.2.	HTTP Method	30
6.4.3.	Input Parameters	30
6.4.4.	Capabilities	31
6.4.5.	Uses	31
6.4.6.	Response	31
6.4.7.	Example	32
7.	IANA Considerations	34
7.1.	Information for IANA on proposed Cost Types	34
7.2.	Information for IANA on proposed Endpoint Properties	34
8.	Acknowledgements	34
9.	References	34
9.1.	Normative References	34
9.2.	Informative References	34
	Authors' Addresses	35

1. Introduction

IETF has designed a new service called ALTO that provides guidance to overlay applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource. This guidance is based on parameters that affect performance and efficiency of the data transmission between the hosts, e.g., the topological distance. The purpose of ALTO 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.

Current ALTO Costs and their modes provide values that are seen to be stable over a longer period of time, such as hopcount and administrative routing cost to reflect ISP routing preferences. Recently, new use cases have extended the usage scope of ALTO to Content Delivery Networks, Data centers and applications that need additional information to select their Endpoints or handle their PIDs.

Thus a multitude of new Cost Types that better reflect the requirements of these applications are expected to be specified, in particular cost values that change more frequently than previously assumed.

The ALTO protocol [ID-alto-protocol] restricts ALTO Cost Maps and Endpoint Cost services to only one Cost Type and Cost Mode per ALTO request. To retrieve information for several Cost Types, an ALTO client must send several separate requests to the server.

It would be far more efficient, in terms of RTT, traffic, and processing load on the ALTO client and server, to get all costs with a single query/response transaction. Vector costs provide a robust and natural input to multi-variate path computation as well as robust multi-variate selection of multiple Endpoints. In particular, 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 transmission. Additionally, for many emerging applications that need information on several Cost Types, having them gathered in one map will save time.

Along with multi-cost values queries, the filtering capabilities need to be extended to allow constraints on multiple metrics. The base protocol allows optional constraints in the input parameters to a

request for a Filtered Cost Map or the Endpoint Cost Service. The 'constraints' member is an AND-combination of expressions that all apply to the (single) requested Cost Type. It is therefore necessary to allow constraints on multiple metrics. Beyond that, applications that are sensitive to several metrics and struggle with complicated network conditions may need to arbitrate between conflicting objectives such as routing cost and network performance. To address this issue, this draft proposes to extend the base protocol by both allowing to combine constraints on multiple metrics and relating these constraints with a logical 'AND' and a logical 'OR'. This allows an application to make compromises such as: "select solutions with either (moderate 'hopcount' AND high 'routingcost') OR (higher 'hopcount' AND moderate 'routingcost')".

This draft is organized as follows: section 3 exposes use cases motivating the introduction of new Cost Types and why multi-cost transactions are useful. Section 4 identifies the core ALTO protocol extensions that are required or recommended to support requests and responses on multiple Cost Types in one single transaction. Section 5 specifies the extended constraints on multi-cost values. Section 6 specifies the protocol extensions for Multi-Cost ALTO transactions and provides examples.

2. Application Scope And Terminology

This draft generalizes the case of a P2P client to include the case of a CDN client, a client of an application running on a virtual server, 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 request router 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 (NP) and reflects its preferences for the choice of endpoints. The NP defines in particular the network map, the routing cost among Network Locations, the cost types used to reflect it, and which ALTO services are available at a given ALTO server.

This draft uses terms defined as follows:

- o Endpoint (EP): can be a Peer, a CDN storage location, a physical server involved in a virtual server-supported application, a Party in a resource sharing swarm such as a computation Grid or an online multi-party game.
- o Endpoint Discovery (EP Discovery) : this term covers the different types of processes used to discover the eligible endpoints.
- o Network Service Provider (NSP): includes both ISPs, who provide means to transport the data, and Content Delivery Networks (CDNs) who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.
- o ALTO transaction: a request/response exchange between an ALTO Client and an ALTO Server.
- o Application Client (AC): this term generalizes the case of a P2P client to include the case of a CDN client, a client of an application running on a virtual server, a GRID application client and any Client having the choice in several connection points for data or resource exchange.

3. Uses Cases For Using Multiple Costs

The ALTO protocol specification in [ID-alto-protocol] focuses on the basic use case of optimizing routing costs in NSP networks. Upcoming use cases however will require both new Cost Types and new Endpoint Properties. Recent ALTO use cases now extend to CDNs, Data centers and other applications that need additional information to select their Endpoints or handle their PIDs. The needed Cost Types depend on the QoE requirements that are specific to the applications. Moreover, the cost values that they may use may change more rapidly than assumed up to now.

The goal of this section is to describe forward looking use case scenarios that are likely to benefit from ALTO, in order to motivate the introduction of new Cost Types and Endpoint Properties as well as the ALTO Multi-Cost extension.

3.1. Use Cases For Using Additional Costs

ALTO Cost Types and Endpoint Properties are registered in 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 ALTO 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 Endpoint 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. New entries should conform to the respective syntactical requirements, and must include information about the new identifier, the intended semantics, and the security considerations. One basic example advocating for multiple Cost Type transactions is an Application Client looking for destination Endpoints or Source/Destination PID pairs yielding jointly the lowest 'routingcost' and path delay. We hereby assume that 'routingcost' values report some monetary cost and that the Application Client chooses to rely on the hopcount to reflect the path delay.

3.1.1. 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.

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 an 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, possibly with respect to the time of day. Also static information like hopcount can serve as an indicator for the delay that can be expected. Thus a Cost Type that can indicate latency, without the need for end-to-end measurements between endpoints, is likely to be useful.

3.1.2. Selection Of Physical Servers Involved In Virtualized Applications

Virtualized applications in large Datacenters are supported by virtualized servers that actually gather resources distributed on several physical servers. The federation of these resources is often orchestrated by a centralized entity that needs to select the physical servers from or to which it will take resources. This

entity can be co-located with an ALTO Client that will request and get the ALTO information on the network formed by the physical servers. The physical servers can be assimilated to endpoints with which the orchestration entity trades application resources or content. These resources include computation resources, storage capacity and path bandwidth between the physical servers.

Here too, the applications that are ran are diverse and may have different and specific QoE requirements. The Endpoint selection typically needs to consider both the computational resources at the Endpoints and the resources e.g. in bandwidth on the transmission paths to or among Endpoints. Thus the application QoE requirements drive the Endpoint selection with more or less weight on QoE specific metrics such as hopcount/delay, bandwidth and other resources, that are typically combined with the routing cost and need to jointly integrate the Endpoint and transmission path perspective in the decision process, which is difficult to do with one single Cost Type.

3.1.3. CDN Surrogate Selection

Another use case is motivated through draft [draft-jenkins-alto-cdn-use-cases-01]. The request router in today's CDNs makes a decision about the surrogate or cache node to which a content request should be forwarded. Typically this decision is based on locality aspects, i.e. the request router tries to select the surrogate node closest to the client. By using the 'routingcost' Cost Type, an ALTO server allows an NSP to guide the CDN in selecting the best cache node. This is particularly important as CDNs place cache nodes deeper into the network (i.e., closer to the end user), which requires finer grained information. Finally the provisioning of abstracted network topology information across administrative boundaries gains importance for cache federations.

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.

For example, information reporting on the occupation level of a cache could be based on a cost reflecting: its remaining computation resources, its remaining storage capacity w.r.t its capacity in storage or computation resources.

As ALTO is likely to become a standardized interface to provide network topology information, the ALTO server could also provide

other information that a request router needs. In the next iterations of this draft we will analyse which of these metrics is suitable as a Cost Type or Endpoint Property for CDN Surrogate Selection, and propose to register them in the respective registries.

3.1.4. Some Proposed Additional Properties And Costs

In addition to CDN caches, Endpoint Properties and Costs can be useful to report an Endpoint's load, given that an Endpoint can as well be a physical server in a datacenter or any entity as defined in Section 2 of this draft.

Proposed new Endpoint properties and costs include:

- o an Endpoint Property called "EP-Capacity", reflecting the nominal capacity of this endpoint. This capacity could be split into:
 - * EP-Nominal-Memory: the storage capacity of the Endpoint.
 - * EP-Nominal-Bandwidth: the capacity of the computation resources of the Endpoint.
- o an Endpoint Cost called "EP-Occupied-Capacity", reflecting the currently available resources w.r.t. their nominal capacity. As with EP-Capacity, this can be split into:
 - * EP-Occupied-Memory: the remaining storage capacity,
 - * EP-Occupied-Bandwidth: the remaining computation resources.

Likewise, new Cost Types are needed to describe the resources of the network paths needed for content transport, in particular the utilized network path bandwidth.

- o A Cost Type named 'pathoccupationcost' (POC) can be used to reflect the NP view of the utilized path bandwidth. Such an ALTO Cost Type is likely to have values that change frequently. By no means, as stated in the ALTO requirements, are ALTO Cost types expected to reflect real-time values, as these can be gathered by other mechanisms. Instead, a Cost Type such as 'pathoccupationcost' should be used as an abstraction that may be represented by a statistical value, or be updated regularly at a frequency lower than 'real-time', or be provided according to different time periods or other parameters. A provision mode for time dependent cost values is proposed in [draft-randriamasy-alto-cost-schedule-01]

3.2. Use Cases For Multi-Cost ALTO Transactions

Different Cost Types are suitable for different applications. For example, delay sensitive applications look for both low routing cost and low delay, where as other applications, such as non real time content download, look for moderate delay and minimal losses. On the other hand, applications or entities managing application input information may want, for various reasons to update their ALTO information on several Cost Types. So an ALTO Client may want to mix Cost Types in either 'numerical' and 'ordinal' mode, for Cost Types values that can be represented by numerical values.

The Multi-Cost ALTO Services propose to:

- o include several Cost Types (and/or Cost Modes) in an ALTO client's Cost Map and Endpoint Cost request,
- o provide several Cost Type values (and/or Cost Mode) in an ALTO server's response, instead of one.

The primary reasons to use Multi-Cost ALTO are:

- o Optimizing time and bandwidth: a single ALTO response with a Multi-Cost cost map with three separate Cost Type values takes much less network bandwidth, and fewer CPU cycles, than three separate ALTO requests for three complete single-cost cost maps. The motivation also holds for the Endpoint Cost Service. Multi-Cost ALTO services can straightforwardly provide a more complete set of cost information.
- o Facing unpredictable and/or rapid value changes: an ALTO client can get a consistent snapshot of several different rapidly-varying Cost Type values.

3.2.1. Optimized Endpoint Cost Service

The Endpoint Cost Service (ECS) provides cost information about both the application Endpoint resources and the networking resources used to access those Endpoints. In addition, the ECS may be invoked in "short term" situations, that is for frequent requests and/or requests requiring fast responses. For the ECS, the server's response is restricted to the requested Endpoints, and so is much smaller than a complete Cost Map. Therefore the ECS can be invoked for 'nearly-instant' information requests, and is particularly well suited for multi-cost ALTO transactions, supporting requests and responses on several Cost Type values simultaneously.

3.2.2. Optimized Filtered Cost Map Service

The set of ALTO Cost Types is not restricted to 'routingcost': ALTO Servers may provide a broader set of metrics. One thing to consider is that the frequency of updates can vary from a Cost Type to another one. Additionally the volume of an entire cost map with values of all available Cost Types, may get rapidly prohibitive for frequent downloads. Given these considerations the Application Client may take better advantage when:

- o requesting multi-cost maps filtered w.r.t. Cost Types of compatible update frequencies or dates, which is the responsibility of the Application Client,
- o requesting multi-cost maps filtered w.r.t. a restricted set of PID pairs.

In such a case, as with the Endpoint Cost Service, the purpose of a Multi-Cost transaction is to gain time with whatever future use of the received ALTO information. In this case, the Client may mix Cost Types in either 'numerical' and 'ordinal' mode, for Cost Type values that can be represented by numerical values.

3.2.3. Cases Of Unpredictable Endpoint Cost Value Changes

Querying all Endpoint cost values simultaneously is always more time and resources efficient than doing it sequentially.

It becomes a necessity in case of unpredictable and/or rapid value changes on at least one of the ALTO Cost Types. The term 'rapid' here means "Typical update intervals [that] may be several orders of magnitude longer than the typical network-layer packet round-trip time (RTT)", as described in [RFC6708], up to a couple of minutes.

This section provides two examples of a delay sensitive application using 'routingcost' and 'hopcount' to select an Endpoint. The application can choose between two candidate Endpoints, EP1 and EP2. The initial choice at T=1 is EP1. It is assumed that at T=2 events in the network occur that impact both 'routingcost' and 'hopcount'.

These examples illustrate the need to query 'hopcount' and 'routingcost' values at the same time in order to re-evaluate the EP costs w.r.t. the QoE needs of the application. It is assumed that the application triggers regular ALTO requests to get the latest cost values for a list of candidate Endpoints.

In some cases the Application client wants to use the ALTO information to perform multi-variate optimization on several Cost

Type values. In order for the optimization to be reliable, it is recommended that the Cost Type values are provided in 'numerical' Cost Mode. Therefore the requested Cost Mode for the applicable Cost Types SHOULD be 'numerical'.

3.2.3.1. Case Of A Multi-Cost ALTO Query Upon A Route Change

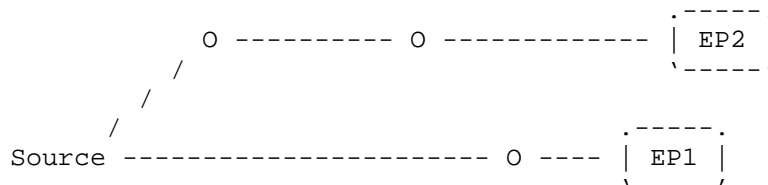
In Figure 1, initially at time T=1, the application has chosen EP1 rather than EP2, despite the higher routing cost, because EP1 has a "better" (lower) 'hopcount' value and despite the higher routing cost and possibly because the application has set a higher weight to 'hopcount'.

At a time T=2, the route to EP1 changes. The ALTO Server information is accordingly updated. The ALTO client makes its next request to update the cost values for 'routingcost' and 'hopcount' on EP1 and EP2. It appears that EP1 has now a hopcount value of 3, the same than for EP2 while its routing cost is higher.

The application realizes that there is no more benefit in keeping interacting with EP1 and therefore switches to EP2, that now has the same hopcount but a lower routing cost.

T = 1 : EP1: routingcost = 40, hopcount = 2
 EP2: routingcost = 30, hopcount = 3

EP1 is selected because application is time-sensitive and
 metric 'hopcount' has a higher weight



T = 2 : EP1: routingcost = 40, hopcount = 3
 EP2: routingcost = 30, hopcount = 3

- Route to EP1 has changed. Hopcount is now 3

=> EP2 is selected because routingcost is lower than for
 EP1, with the same hopcount value

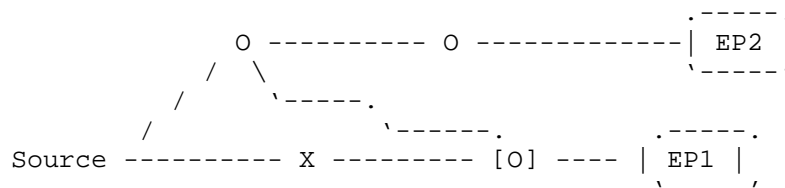
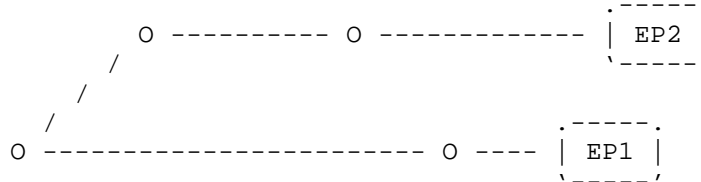


Figure 1: Endpoint re-selection using Multi-Cost ALTO request on
 updated cost values, upon a change in the route.

3.2.3.2. Case Of A Multi-Cost ALTO Query Upon A Cost Value Change

T = 1 : EP1: routingcost = 30, hopcount = 2
 EP2: routingcost = 30, hopcount = 3
 ==> EP1 is selected because application is time-sensitive and
 hopcount metrics has higher weight



T = 2 : EP1: routingcost = 40, hopcount = 2
 EP2: routingcost = 30, hopcount = 3
 Routingcost to EP1 has increased. Hopcount is the same.
 ==> Delay sensitive applications willing to minimize hopcount
 remain with EP1 while other applications may remain
 with EP2, that now has a lower routingcost.

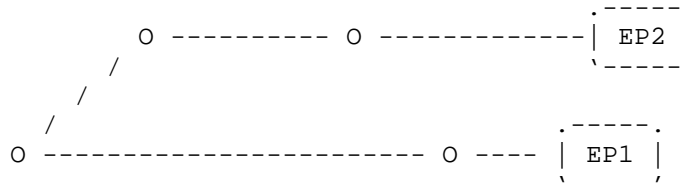


Figure 2: Endpoint selection using 2 Cost Types with joint request on updated cost values and for delay sensitive applications.

4. ALTO Protocol Updates Needed To Support Multi-Cost Transactions

To allow running Multi-Cost ALTO Services some minor changes in the base protocol are needed. A set of multi-cost specific media-types is introduced and the main updates consist of changing the JSON type of the value taken by a few members of the objects describing the information resources.

As written in the introduction, this section relies on Section {11.2.3.6} of the ALTO protocol draft, see [ID-alto-protocol], which allows protocol extensions to encode cost values as the 'JSONValue' data type.

4.1. List Of ALTO Protocol Updates Required And Recommended

The following updates to the ALTO protocol ([ID-alto-protocol]) are required or recommended to support multi-cost ALTO transactions. The new resulting JSON formats are specified in the next sections. Section references ({##}) are to the ALTO protocol document.

- o Updates required in the format of objects member(s):
 - * Objects DstCosts (to destination PIDs, {11.2.3.6}) and EndpointDstCosts (to destination Endpoints, {11.5.1.6}): JSON type of cost value member evolves from JSONNumber to JSONArray.
 - * Object ReqFilteredCostMap {11.3.2.3} and ReqEndpointCostMap {11.5.1.3}: member cost-type be redefined as an array of CostMode values, rather than a single value.
- o Updates recommended in the object structure:
 - * The capabilities for the Filtered Cost Map Service {11.3.2.4} and the Endpoint Cost Map Service {11.5.1.4} be extended with a new member giving the maximum number of Cost Types allowed in a request.
- o Rules required on object member description:
 - * Order in which the multiple cost values are provided in the responses,
 - * Number of values in member 'cost-types' of objects InfoResourceCostMap, InfoResourceEndpointCostMap, ReqFilteredCostMap, ReqEndpointCostMap.
- o Rule recommended on the cost value mode:
 - * when the mode 'numerical' is available or applicable.

4.2. Updates Required In The Member Format Of Objects

This section specifies the changes in the object member format that are required to enable multi-cost ALTO transactions.

The term Single Cost qualifies the items as they are specified in the current ALTO protocol.

4.2.1. Cost Value Encoded In JSONArray

The fundamental change to support multi-cost is to encode the cost values with the type JSONArray. This way, the cost between two PIDs or two Endpoints can be represented in a generic way:

- o with several Cost Types,
- o with Cost Types whose value can each be encoded with any type of JSON value.

For example, a multi-cost value represented with Cost Types (assuming they are supported by the ALTO Server):

```
["num-routingcost", "num-hopcount", "calendar-quarterlyvalue", "string-status"]
```

will be encoded in the following JSON Array in a Multi Cost ALTO response:

```
[23, 6, [2, 5, 4, 1], "medium"]
```

The objects impacted by the encoding of ALTO Multi-Cost values in a JSONArray are: DstCosts and EndpointDstCosts. Full specification will be provided in later sections of this draft.

4.2.2. Scalar 'cost-type' Member Replaced By Array 'cost-types' Member

In the base protocol, the various single-cost-map services use a scalar "cost-type" member in the "meta" section to indicate the cost metric and cost mode of the returned values.

In Multi-Cost ALTO, the multi-cost-map services use an array member named "cost-types" instead. The array elements are in the same format as the "cost-type" member in single cost maps, and the order corresponds to the order of values in the array values in the multi-cost map.

Alternatively, we could use the same member name, but define it as an array for multi-cost services. This would simplify some things for a client, but complicate others. Overall, we believe it is easier for a client to use a new member name than to overload the type of an existing member name.

4.2.3. Rule On Cost Value Order In ALTO Responses

The cost values each Source/Destination pair MUST be provided in the same order as in the array of Cost Types. This way, the cost type values are provided without any ambiguity on the Cost Type they report on.

4.3. Updates Recommended In The Object Structure

Objects `MultiCostMapCapability` and `FilteredMultiCostMapCapability`: new member giving the maximum number of Cost Types in a response.

5. Extended Constraints On Multi-Cost Values

This draft proposes to extend the constraint tests in the base protocol to allow tests on the various costs in a request, and to allow more general predicates.

NOTE: Constraint tests on multiple cost metrics are useful even when retrieving single costs, and we expect there will be proposals to add multi-cost constraint tests to the ALTO protocol, relating to the extensions proposed in this draft. Draft [draft-lee-alto-app-net-info-exchange] proposes in particular extensions to query values on a metric M1 with constraints on other metrics M2, ... Mk, that adds an interesting feature to extend ALTO constraints.

The base ALTO protocol allows optional constraints in the input parameters to a request for a Filtered Cost Map or the Endpoint Cost Service. The 'constraints' member is an array of expressions that all apply to the (single) requested Cost Type. The encoding of 'constraints' member, is fully specified in Section 11.3.2.3 of the base protocol as follows:

- A constraint contains two entities separated by whitespace:
 - (1) an operator, 'gt' for greater than, 'lt' for less than, 'ge' for greater than or equal to, 'le' for less than or equal to, or 'eq' for equal to
 - (2) a target cost value. The cost value is a number that MUST be defined in the same units as the Cost Type indicated by the `costtype` parameter

...

If multiple 'constraint' parameters are specified, they are interpreted as being related to each other with a logical AND.

Such a specification covers multiple predicates on one metric such as:

'routingcost' values belong to [6, 20)

5.1. Use Cases For Multi-Cost Multi-Operator Constraints

Suppose that an application uses information on the ALTO Cost Types 'hopcount' and 'routingcost'. This application may want to select paths or Endpoints with bounds on values for both 'hopcount' and 'routingcost'. For instance solutions meeting a constraint like:

'hopcount' values in [6,20) OR 'routingcost' values in [100,200]

Moreover, this application may be ready to make compromises and to select paths or Endpoints by bounding their cost values according to two options:

1. either solutions with moderate 'hopcount' and high 'routingcost', for instance: 'hopcount' values in [6,20] AND 'routingcost' values in [100,200],
2. or solutions with higher 'hopcount' and moderate 'routingcost', for instance: 'hopcount' values in [20,50] AND 'routingcost' values in [30,100].

5.2. Extended constraints in Multi-Cost ALTO

This draft proposes to support the two above mentioned use cases by extending the scope of constraints in two ways:

- o allow the 'constraint' member to be applicable to multiple Cost Types,
- o allow the multiple constraints to be related to each other by both logical AND and logical OR.

The two options would be covered by a logical expression like:

```
[('hopcount' ge 6) AND ('hopcount' lt 20) AND  
( 'routingcost' ge 100) AND ('routingcost' le 200)]  
OR  
[( 'hopcount' ge 20) AND ('hopcount' le 50) AND  
( 'routingcost' ge 30) AND ('routingcost' le 100)]
```

A simple encoding of multi-cost constraints for such expressions is specified in Section 5.3.3 of this draft, describing the input parameters to request for Filtered Cost Map. This specification is applicable to the EP Cost service as well.

6. Protocol Extensions For Multi-Cost ALTO Transactions

This section proposes extensions 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 information on several Cost Types, then 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 an array, where each component corresponds to a specified Cost Type. The correspondence between the components and the Cost Types is implicitly indicated in the ALTO response. Indeed, the values in the Cost values MUST be provided in the same order as in the array of cost types indicated in the response.

The following ALTO services have corresponding Multi-Cost extensions:

- o Information Resources Directory: extended with multi-cost related URIs and associated capabilities.
- o Cost Map Service: extended with the Multi-Cost Map Service,
- o Cost Map Filtering Service: extended with the Multi-Cost Map Filtering Service,
- o Endpoint Cost Lookup Service: extended with the Endpoint Multi-Cost Lookup Service.

6.1. Information Resources Directory

When the ALTO server supports the provision of information on multiple costs in a single transaction, the Information Resources Directory will list the corresponding resources. The media type remains the same as in the current ALTO protocol.

6.1.1.1. Example of Multi-Cost specific resources in the IRD

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 additionally provides 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.

In this example, the ALTO IRD exposes Multi-Cost capabilities on cost types "routingcost", "hopcount", "pathoccupationcost", that can be combined in a request. The values on these metrics are provided in numerical mode. Values provided for cost-type string are in "string" mode.

```
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
```

```
{
  "meta" : {
    "cost-types" : {
      "num-pathoccupationcost" : {
        "cost-mode" : "numerical",
        "cost-metric" : "pathoccupationcost"
      },
      "str-status" : {
        "cost-mode" : "string",
        "cost-metric" : "status"
      },
      "num-routing" : {
        "cost-mode" : "numerical",
        "cost-metric" : "routingcost"
      },
      "num-hopcount" : {
        "cost-mode" : "numerical",
        "cost-metric" : "hopcount"
      },
      .....
      Other ALTO cost types as described
    }
  }
}
```

```

        in current ALTO Protocol
        .....
    },
    "default-alto-network-map" : "my-default-network-map"
},
"resources" : {
    "my-default-network-map" : {
        "uri" : "http://alto.example.com/networkmap",
        "media-type" : "application/alto-networkmap+json"
    },
    "numerical-routing-cost-map" : {
        .....
        Single-cost Services as described
        in current ALTO Protocol
        .....
    },
    "routingcost-hopcount-multicost-map" : {
        "uri" : "http://alto.example.com/multi/costmap",
        "media-types" : [ "application/alto-multicostmap+json" ],
        "uses" : [ "my-default-network-map" ],
        "capabilities" : {
            "cost-type-names" : [ "num-routing", "num-hopcount" ]
        }
    },
    "filtered-multicost-map" : {
        "uri" : "http://alto.example.com/multi/costmap/filtered",
        "media-types" : [ "application/alto-multicostmap+json" ],
        "accepts" : [ "application/alto-multicostmapfilter+json" ],
        "uses" : [ "my-default-network-map" ],
        "capabilities" : {
            "cost-constraints" : true,
            "max-cost-types" : 3,
            "cost-type-names" : [ "num-routingcost",
                                "num-hopcount",
                                "num-pathoccupationcost" ]
        }
    },
    "endpoint-multicost-map" : {
        "uri" : "http://alto.example.com/multi/endpointmulticost/lookup",
        "media-types" : [ "application/alto-endpointmulticost+json" ],
        "accepts" : [ "application/alto-endpointmulticostparams+json" ],
        "uses" : [ "my-default-network-map" ],
        "capabilities" : {
            "cost-constraints" : true,
            "max-cost-types" : 2,
            "cost-type-names" : [ "num-routingcost",
                                "num-hopcount",
                                "str-status" ]
        }
    }
}

```



```
    }  
  }  
}
```

6.2. 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 values of the different Cost Types 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.

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 as a JSONArray of cost values for the different Cost Types.

Note that values in a Multi-Cost map are arrays of values of the various Cost Types. If the ALTO server does not have the value for a particular Cost Type for a source/destination PID pair, the server MUST use 'null' (a reserved JSON symbol) for that location in the array. If the ALTO server does not have a value for any of the Cost Types for a given source/destination pair -- that is, if the array would be a list of nulls -- then the ALTO server MAY omit the array for that source/destination pair.

6.2.1. Media Type

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

6.2.2. HTTP Method

This resource is requested using the HTTP GET method.

6.2.3. Input Parameters

None.

6.2.4. Capabilities

The capabilities of the URI providing this resource are defined by a JSON object of type `FilteredCostMapCapabilities::`

```
object {  
    JSONString cost-type-names<1..*>;  
} MultiCostMapCapabilities;
```

with members

cost-type-names The Cost Type names returned by this map.

An ALTO Server MUST support all of the Cost Types listed here. 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.

6.2.5. Uses

The Resource ID of the Network Map which defines the PIDs used in this Multi Cost Map. An ALTO Server MUST NOT define two Multi Cost Maps with the same Network Map and set of Cost Types.

6.2.6. Response

The "meta" field of a Cost Map response MUST include the "dependent-tags" key, whose value is a single-element array to indicate the Version Tag of the Network Map used, where the Network Map is specified in "uses" of the IRD.

The "meta" MUST also include the member "cost-types", which is a JSONArray of the CostTypes in this Multi Cost Map.

The data component of a Multi Cost Map response is named "multi-cost-map", which is a JSON object of type CostMapData, as defined in {11.2.3.6} of the ALTO protocol. This is identical to the format of the ALTO Cost Map response, except that the JSONValues are arrays rather than numbers. The values in the arrays correspond to the Cost Type listed at the same place in the 'cost-types' array. This array MUST have the same size as the 'cost-types' array, and the values in the MUST be in the same order as in the 'cost-types' array.

The returned Multi Cost Map MUST include the required Path Costs for each pair of Source and Destination PID for which this information is available. If a cost value is not defined, the ALTO Server MUST replace that value in the array with the reserved JSON symbol 'null'. If no costs are defined for a pair of Source and Destination PIDs, so

the Path Cost would be an array of nulls, the ALTO Server MAY omit the array for that pair.

6.2.7. Example

This example illustrates a 'static' multi-cost ALTO transaction, 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 report on the path delay. We also assume that ALTO server does not know the value of the "routingcost" between PID2 and PID3, and hence uses null for those costs.

```
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" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "hopcount" }
    ]
  }
  "multi-cost-map" : {
    "PID1": { "PID1": [1,0], "PID2": [5,23], "PID3": [10,5] },
    "PID2": { "PID1": [null,5], "PID2": [1,0], "PID3": [15,9] },
    "PID3": { "PID1": [20,12], "PID2": [null,1], "PID3": [1,0] }
  }
}
```

6.3. Filtered Multi-Cost Map

A Multi-Cost Map may be very large. In addition, an Application Client assisted by the ALTO Client does not necessarily need the Cost Types for all the source/destination PID pairs.

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

6.3.1. Media Type

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

6.3.2. HTTP Method

This resource is requested using the HTTP POST method.

6.3.3. 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 ReqFilteredMultiCostMap, where:

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

object {
  CostType    cost-types<1..*>;
  JSONString  constraints<0..*>;      [OPTIONAL]
  JSONArray   or-constraints<0..*>;   [OPTIONAL]
  PIDFilter    pids;                  [OPTIONAL]
} ReqFilteredMultiCostMap;
```

with members:

cost-type The Cost Types for the returned costs.

Each listed Cost Type MUST be one of the supported Cost Types indicated in this resource's capabilities.

constraints

Defines an array of additional constraints. The ALTO server MUST return the costs for all source/destination pair that satisfy all constraints in this list, and no other

costs. This parameter MUST NOT be specified if this resource's capabilities indicate that constraint support is not available. Each string in the 'constraint' array MUST contain three entities separated by whitespace, in the following format:

[index] op value

'Index' is a number between 0 and the number of Cost Types minus 1, and indicates the Cost Type to which this constraint applies. (The square brackets ([]) surrounding 'index' are required syntactic sugar. They serve as a reminder that 'index' is an array index, not a value to test, and they avoid unusual-looking constraints such as "1 ge 5".) 'Op' is an operator: 'gt' for greater than, 'lt' for less than, 'ge' for greater than or equal to, 'le' for less than or equal to, 'eq' for equal to, or 'ne' for not equal to. 'Value' is a target cost value to compare against the indicated Cost Type. For numeric Cost Types, 'value' MUST be a number defined in the same units as the Cost Type indicated by 'index'. ALTO servers SHOULD use at least IEEE 754 doubleprecision floating point [IEEE.754.2008] to store the cost value, and SHOULD perform internal computations using double-precision floating-point arithmetic. For string Cost Types, 'value' MUST be a string enclosed in single quotes ('). For array-valued Cost Types, 'eq' is true iff one of the Cost Type values is equal to 'value', and 'ne' is true iff none of the Cost Type values are equal to 'value'. The other operators are not defined for array-valued Cost Types.

or-constraints

Defines an array of arrays of constraint strings. The individual constraint strings MUST be in the same format as strings in the 'constraints' parameter. The ALTO server MUST return costs that satisfy all constraints in one or more of the inner lists, and no other costs. That is, 'or-constraints' is the logical OR of ANDs. The client MUST NOT specify this parameter if this resource's capabilities indicate that constraint support is not available. The client MUST NOT specify both a 'or-constraints' and a 'constraints' parameter.

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.

6.3.4. Capabilities

The URI providing this resource supports all capabilities documented in Section 6.2.4 (with identical semantics), plus additional capabilities. In particular, the capabilities are defined by a JSON object of type `FilteredMultiCostMapCapability`:

```
object {  
  CostType    cost-types<1..*>;  
  JSONBool    cost-constraints;  
  JSONNumber  max-cost-types; [OPTIONAL]  
} FilteredMultiCostMapCapability;
```

with members:

`cost-types` The cost types available from this service.

`max-cost-types` Indicates the maximum number of cost values the ALTO Server can provide in a multi-cost array of a Multi-Cost Map.

`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.

6.3.5. Uses

The Resource ID of the Network Map which defines the PIDs used in this Filtered Multi Cost Map.

6.3.6. Response

The response is the same format as for the Multi Cost Map Service (Section 6.2.6). 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.

6.3.7. Example 1

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

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

```
{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "hopcount" }
    ]
  },
  "multi-cost-map" : {
    "PID1": { "PID1": [1,6], "PID2": [5,23], "PID3": [10,5] }
  }
}
```

6.3.8. Example 2

This is an example of using constraints to restrict returned source/destination PID pairs to those with 'routingcost' between 5 and 10, or 'hopcount' equal to 0.

```
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-types" : [
    { "cost-mode": "numerical", "cost-metric": "routingcost" },
    { "cost-mode": "numerical", "cost-metric": "hopcount" }
  ],
  "or-constraints" : [ [ "[0] ge 5", "[0] le 10" ],
                       [ "[1] eq 0" ] ],
  "pids" : {
    "srcs" : [ "PID1", "PID2" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}
```

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

```
{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "hopcount" }
    ]
  }
  "multi-cost-map" : {
    "PID1": { "PID2": [5,23], "PID3": [10,5] },
    "PID2": { "PID2": [1,0] },
  }
}
```

6.4. Endpoint Multi-Cost Service

The Endpoint Multi-Cost Service provides information on several Cost Types between individual Endpoints.

This service 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 costs between the PIDs corresponding to the endpoints if these values are available for the requested Cost Types.

When the cost values are requested to perform multi-variate numerical optimization and are each available in the 'numerical' mode, then the ALTO Client SHOULD request the 'numerical' mode in order to get a reliable result. Note that this consideration is outside the scope of the ALTO protocol as it relates to the responsibility of the ALTO Client and related entries. However common sense lead to warn that a necessary condition for vector ranking method to be reliable is that the components of the processed vectors are numerical and not ordinal values.

6.4.1. Media Type

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

6.4.2. HTTP Method

This resource is requested using the HTTP POST method

6.4.3. 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-endpointmulticostparams+json", which is a JSON Object of type ReqEndpointMultiCostMap:

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

object{
  CostType      cost-types<1..*>;
  JSONString    constraints<0..*>;      [OPTIONAL]
  JSONArray     or-constraints<0..*>;   [OPTIONAL]
  EndpointFilter endpoints;
} ReqEndpointMultiCostMap;
```

with members:

cost-types Defined equivalently to the "cost-types" input parameter of a Filtered Multi Cost Map.

constraints Defined equivalently to the "constraints" input parameter of a Filtered Multi Cost Map.

or-constraints Defined equivalently to the "or-constraints" input parameter of a Filtered Multi Cost Map.

endpoints A list of Source Endpoints and Destination Endpoints for which Path multiple 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.

6.4.4. Capabilities

The capabilities are the same as described in Section 6.3.4.

6.4.5. Uses

As with the ALTO Endpoint Cost Service, the Endpoint Multi Cost Service MUST NOT use a Network Map.

6.4.6. Response

The "meta" field of an Endpoint Multi Cost response MUST include the "cost-types" key, to indicate the Cost Types used.

The data component of an Endpoint Multi Cost response is named "endpoint-multi-cost-map", which is a JSON object of type EndpointCostMapData, as defined in Section 11.5.1.6 of the ALTO protocol. This is identical to the format of the ALTO Cost Map response, except that the JSONValues are arrays rather than numbers. The values in the arrays correspond to the Cost Type listed at the same place in the 'cost-types' array. This array MUST have the same size as the 'cost-types' array, and the values in the MUST be in the same order as in the 'cost-types' array.

6.4.7. Example

```
POST multi/endpointmulticost/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-types" : [
    { "cost-mode": "numerical", "cost-metric": "routingcost" },
    { "cost-mode": "numerical", "cost-metric": "hopcount" }
  ],
  "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" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      }
    ],
    "cost-types" : [
      { "cost-mode": "numerical", "cost-metric": "routingcost" },
      { "cost-mode": "numerical", "cost-metric": "hopcount" }
    ]
  }
  "endpoint-multi-cost-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]
    }
  }
}
```

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 Properties

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

The authors would like to thank Dave Mac Dysan and Vijay Gurbani for fruitful discussions and comments on this draft and previous versions.

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-protocol]
"ALTO Protocol" draft-ietf-alto-protocol-27.txt", March 2014.

- [RFC6708] "Application-Layer Traffic Optimization (ALTO) Requirements", February 2012.
- [draft-jenkins-alto-cdn-use-cases-01]
"Use Cases for ALTO within CDNs" draft-jenkins-alto-cdn-use-cases-01", June 2011.
- [draft-lee-alto-app-net-info-exchange]
"Information Exchange for High Bandwidth Applications in TE networks", October 2013.
- [draft-randriamasy-alto-cost-schedule-01]
"ALTO Cost Schedule", July 2012.
- [draft-randriamasy-alto-multi-cost-05]
"Multi-Cost ALTO", October 2011.

Authors' Addresses

Sabine Randriamasy
Alcatel-Lucent/Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

Wendy Roome
Alcatel-Lucent/Bell Labs
600 Mountain Ave, Rm 3B-324
Murray Hill, NJ 07974
USA

Phone: +1-908-582-7974
Email: w.roome@alcatel-lucent.com

Nico Schwan
Thales Deutschland
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: nico.schwan@thalesgroup.com

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: January 4, 2015

W. Roome
Alcatel-Lucent
Y. Yang
Yale
July 3, 2014

PID Property Extension for ALTO Protocol
draft-roome-alto-pid-properties-02

Abstract

This document extends the Application-Layer Traffic Optimization (ALTO) Protocol [I-D.ietf-alto-protocol] by defining PID-based properties in much the same way that the original ALTO Protocol defines endpoint-based properties.

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 4, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. The Consistency and Inheritance Design Views	3
3. A Hierarchical View of a Network Map	3
3.1. Default Containment Hierarchy	3
3.2. Extension: Implicit Inheritance Via Nested PIDs	4
4. Services	5
4.1. PID Properties Announcement	5
4.2. Full PID Property Map Service	6
4.3. Filtered PID Property Map Service	7
4.4. Endpoint Property Service	7
5. Security Considerations	7
6. IANA Considerations	8
7. References	8
Authors' Addresses	8

1. Introduction

A key abstraction introduced by the ALTO Protocol [I-D.ietf-alto-protocol] is PIDs (Provider-defined Identifiers), where each PID is defined as a name and a set of associated endpoint addresses. For IPv4/IPv6 networks, a PID's address set is defined by one or more endpoint address prefixes called CIDRs [RFC.4632]. This extension focuses on IPv4/IPv6 networks.

An ALTO Server uses PIDs when defining one or more Network Maps, each of which is defined by a set of PIDs. Each Network Map defines a logical partition of a network address space, where similar endpoints are grouped in the same PID, specified by the addresses contained in the definition of the PID. An ALTO Server may publish multiple Network Maps when there are multiple ways to partition networks. For example, one Network Map may partition endpoints according to geographical locations, and hence each PID defined in the Network Map represents the set of endpoints at a given location. Another Network Map may partition endpoints according to the capabilities (e.g., CDN delivery protocols such as HTTP or HTTPS) that the network can provide. In this case, each PID defined in the Network Map represents the endpoints with similar capabilities.

A major missing component of the base ALTO Protocol is that the common properties are not specified. In particular, in the base ALTO Protocol, each PID has only a name and a set of endpoint addresses. The objective of this document is to allow PIDs to have properties. Example PID properties include "country code", "continent code", "ISP", "lat/long bounding box", "endpoint type" (server farm, end users, cell data connections, etc). We identify use cases (e.g., VPN

selection and CDN Capability Advertisement) where PID properties can provide value.

2. The Consistency and Inheritance Design Views

When we define PID properties, we follow a key consistency design guideline that PID properties should be consistent with and generalize the endpoint properties already defined in the base ALTO Protocol. Specifically, in the base ALTO Protocol, for each selected endpoint address, there can be a set of (prop-type, value) pairs associated with the endpoint address. These are called the endpoint properties of the selected endpoint. The ALTO Protocol allows an ALTO Client to obtain defined endpoint properties.

Consider a given endpoint property p and all endpoints defined in a PID named $pid1$. If all of the endpoints have the same value v for p , then it is natural and consistent that when we define the value for p , as a PID property, the value should be v . For the more general case, let $ip1.p$ denote the value of property p for endpoint $ip1$. Assume that $pid1$ consists of a set of n IP addresses, $ip1, ip2, \dots, ipn$. Let $pid1.p$ denote the value of property p for $pid1$. Then we can consider that $pid1.p$ is from an aggregation function of $ip1.p, ip2.p, \dots, ipn.p$. Example aggregation functions include average/mean, mode, geo-center, union, bounding box, where meaningful aggregations depend on the specific property p .

Complementing the bottom-up aggregation view, we also adopt a top-down inheritance view, by considering that when $ip1$ is in $pid1$, $ip1.p$ inherits the value of $pid1.p$, if the value of $ip1.p$ is not defined; otherwise, $ip1.p$ overrides the value of $pid1.p$. The concept of inheritance is a simple, but powerful concept to reduce information redundancy.

3. A Hierarchical View of a Network Map

3.1. Default Containment Hierarchy

A Network Map defined in the base ALTO Protocol can be considered as a default three-level hierarchy: with the highest (1st) level being a root, the next (2nd) level being the PIDs, and the lowest (3rd or leaf) level being the individual endpoint addresses. An issue that the base ALTO Protocol needs to resolve is that PID definitions can overlap, and hence we must determine the PID to which an endpoint address belongs. For example, consider a Network Map with two PIDs: $PID1$ is $10.0.0.0/8$, and $PID2$ is $10.0.1.0/24$. Then all addresses in $PID2$ are also in $PID1$. The base ALTO Protocol requires that an endpoint address be in one, and only one, PID, among the set of PIDs defined in the same Network Map. ALTO achieves this by specifying

that if an address matches several CIDR, the address is in the PID with the CIDR with the longest prefix. We refer to this PID as the home PID of the endpoint. Thus, for the example, 10.0.1.5 is in PID2, and 10.0.2.6 in in PID1.

3.2. Extension: Implicit Inheritance Via Nested PIDs

We would like to use the PID hierarchy to inherit property values. That is, if all endpoints in *px*, *py* and *pz* are also in *pa*, then unless otherwise overridden, PIDs *px*, *py* and *pz* should inherit all properties defined in PID *pa*.

Unfortunately overlapping PID definitions result in the usual issues with multiple inheritance. Consider the following example:

```
PID p1: [1.0.0.0/8]
PID p2a: [1.0.0.0/16]
PID p2b: [1.1.0.0/16]
PID p3: [1.0.255.0/24, 1.1.0.0/24]
```

All endpoints in *p2a* and *p2b* are also in *p1*, so those two PIDs should inherit any properties defined in *p1*. However, the endpoints in *p3* are split between *p2a* and *p2b*, so *p3* cannot simply inherit values from *p2a* or *p2b*. On the other hand, all endpoints in *p3* are in *p1*, so we would expect *p3* to inherit any properties defined in *p1* that are not overridden in *p2a* or *p2b*.

Hence we will define inheritance as follows.

Definition: The immediate parent of CIDR *C* is the CIDR *C'* with the longest prefix of all CIDRs, in the set of all CIDRs in all PIDs in the Network Map, which contain all endpoints in *C*. The immediate parent CIDR might not exist, but if it does, it is unique.

Definition: A CIDR *C* inherits the value *V* for property *PR* if the PID containing its immediate parent CIDR *C'* defines the value *V* for property *PR*, or if its immediate parent CIDR *C'* inherits the value *V* for property *P*.

Definition: A PID *P* has the value *V* for property *PR* if that value is explicitly defined for *PR* in *P*, or if all CIDRs *C* in *P* inherit the same value *V* for property *PR*.

Suppose the following properties are defined for PIDs described above:

```
PID p1:  ISP="Verizon" country-code="us"  
PID p2a: state-code="NJ"  
PID p2b: state-code="NY"
```

Then p2a, p2b, and p3 would all inherit the ISP and country-code properties from p1. However, p3 would not inherit the state-code property, because it has different values in p2a and p2b.

4. Services

In the interests of simplicity, we will give an overview of the proposed services, rather than detailed descriptions.

4.1. PID Properties Announcement

Given the consistency and inheritance design guideline, we require that PID Properties and Endpoint Properties use the same property name space. Such property names must be registered with IANA.

To allow an ALTO Client to know the set of PID Properties associated with a PID Property Resource, we use the same approach as that of endpoint properties: announcement in IRD. An example is shown below.

```

...
"resources" : {
  "my-default-network-map" : {
    "uri" : "http://alto.example.com/networkmap",
    "media-type" : "application/alto-networkmap+json"
  },
  "endpoint-property" : {
    "uri" : "http://alto.example.com/endpointprop/lookup",
    "media-type" : "application/alto-endpointprop+json",
    "accepts" : "application/alto-endpointpropparams+json",
    "capabilities" : {
      "prop-types" : [ "my-default-network-map.pid",
                       "priv:ietf-example-prop" ]
    },
  },
  "my-pid-property" : {
    "uri" : "http://alto.example.com/pidprop/netmap1/pidp1",
    "media-type" : "application/alto-pidprop+json",
    "uses" : [ "my-default-network-map" ]
    "capabilities" : {
      "prop-types" : [ "country-code",
                       "asn" ]
    },
  },
}
}

```

4.2. Full PID Property Map Service

Analogous to ALTO's Full Cost Map Service, a Full PID Map Service returns properties defined for all PIDs in a Network Map.

This is a GET request. The response message is similar to that of ALTO's Endpoint Property Service, but with PID names instead of endpoint addresses. The IRD entry for the service defines a "prop-types" capability with the names of the properties that this service returns, and specifies a "uses" attribute for the Network Map defining the PIDs.

In the interests of limiting the response message size, the Full PID Property Map Service would NOT enumerate inherited property values. Thus if PID1 defines PROP1, and if PID2 is contained within PID1 and does not override the value for PROP1, then the response message gives a value for PROP1 in PID1, but not in PID2. In this case the client is expected to deduce the inheritance. That is feasible because the client has all information needed to do that.

4.3. Filtered PID Property Map Service

Analogous to ALTO's Filtered Cost Map Service, a Filtered PID Map Service returns a subset of the Full PID Property Map. The client specifies the desired property and PID names.

This is a POST request. The response message is the same as for the Full PID Property Map Service. The request message is similar to the request message for ALTO's Endpoint Property Service, except with PID names instead of endpoint addresses. The IRD entry for the service defines a "prop-types" capability with the names of the properties this service returns, and specifies a "uses" attribute for the Network Map defining the PIDs.

Unlike the Full Filtered PID Property Service, the Filtered PID Property Service would explicitly enumerate inherited property values. Thus if PID1 defines PROP1, and if PID2 is contained within PID1 and does not override the value for PROP1, then the response message includes PID1's value for PROP1 in PID2's properties. This is necessary because the Filtered PID Property Map response does not give the client enough information to deduce the inherited properties. For consistency, the Filtered PID Property Service would enumerate inherited properties for a PID even if the client also requested properties for all PIDs that containing that PID.

4.4. Endpoint Property Service

As described in Section 10.8 of the ALTO protocol specification, endpoint property names may be prefixed with the Resource ID of a Network Map. For such resource-specific properties, if a value is not explicitly defined for an endpoint, the Endpoint Cost Service MUST return the value that the Filtered PID Property Map Service would return for the PID containing that endpoint.

For properties that are not prefixed by a Network Map Resource ID, if a value is not defined for an endpoint, the Endpoint Property Service MAY return the value defined for that property in one of the ALTO Server's PID Property Maps for the PID containing the endpoint.

5. Security Considerations

Some properties may have sensitive customer-specific information. If this is the case, an ALTO Server may limit access to those properties by providing several different PID property services. For non-sensitive properties, the ALTO Server would provide a uri which accepts requests from any client. Sensitive properties, on the other hand, would only be available via a secure uri which would require client authentication.

6. IANA Considerations

No actions are required from IANA as result of the publication of this document.

7. References

[I-D.ietf-alto-protocol]

Almi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-20 (work in progress), October 2013.

[RFC.4632]

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

Authors' Addresses

Wendy Roome
Alcatel-Lucent/Bell Labs
600 Mountain Ave, Rm 3B-324
Murray Hill, NJ 07974
USA

Phone: +1-908-582-7974
Email: w.roome@alcatel-lucent.com

Y. Richard Yang
Yale University
51 Prospect St.
New Haven, CT
USA

Email: yry@cs.yale.edu

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2015

M. Scharf
Alcatel-Lucent Bell Labs
July 2, 2014

Path-Vector and Node-Edge Topology Maps in ALTO
draft-scharf-alto-topology-00

Abstract

The Application-Layer Traffic Optimization (ALTO) service defines network and cost maps to provide basic network information. This document motivates an optional extension of ALTO for the definition of additional topology properties. The ALTO extension supports both path-vector and node-edge topology maps.

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 3, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Endpoint-Centric Network Topology Models	3
3.1. Use Cases	3
3.2. Topology Models	4
4. Topology Encoding Formats for ALTO	5
4.1. Map Service Extension	5
4.2. Path-Vector Format	6
4.3. Node-Edge Format	8
5. IANA Considerations	10
6. Security Considerations	10
7. Conclusion	10
8. References	10
8.1. Normative References	10
8.2. Informative References	10
Appendix A. Contributors	11
Author's Address	11

1. Introduction

The Application-Layer Traffic Optimization (ALTO) service [I-D.ietf-alto-protocol] provides network information (e.g., basic network location structure and 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 a simplified view, yet enough information about a network for applications to effectively utilize them.

Applications using an ALTO service typically consist of instances running at endpoints. This is why ALTO maps provide abstract cost and/or ranking information between network endpoints. Yet, for selected use cases of ALTO, it is desirable to have a more detailed representation of the network. For instance, in settings with multiple application source-destination pairs with multiple links, such a representation could help avoid bottleneck or failed links.

Topology models for ALTO are also discussed in other related documents. A full solution to extend ALTO is defined in [I-D.yang-alto-topology]. An earlier survey of use-cases for extended network topology information can also be found in [I-D.bernstein-alto-topo]. Further related work has been published in [I-D.lee-alto-app-net-info-exchange] and [I-D.bernstein-alto-large-bandwidth-cases].

This document specifies two graph representation formats that can be used in ALTO. Both formats are optional, and it is up to the operator of an ALTO service to decide whether to offer this data in addition to the standard network and cost maps of an ALTO service. The graph representation defined in this document is based on existing ALTO abstraction (e.g., PIDs) and complements the existing path-based ALTO cost map representation. Together, they provide a more complete but still abstract representation of networks for informed traffic optimization among endpoints.

This document does not intend to model topology internals not affecting endpoints, such as routing protocol internals. A data model for network topologies with routing protocol externals can for instance be found in [I-D.clemm-i2rs-yang-network-topol].

2. Terminology

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

3. Endpoint-Centric Network Topology Models

3.1. Use Cases

This document focuses on exposure of network topology properties that matter to endpoints. Usually, applications instances at network endpoints (hosts) only have to care about end-to-end network properties of the paths between those endpoints. The basic ALTO services, i.e., either the Network and Cost Map service or the Endpoint Cost service, can be used to express end-to-end characteristics of paths between endpoints, as explained e.g. in [I-D.ietf-alto-deployments] and [I-D.wu-alto-te-metrics].

However, there are a number of use cases in which it is difficult to expose topological properties to applications using only the mechanisms in the base ALTO protocol [I-D.ietf-alto-protocol]. These uses cases are typically characterized by multiple application source-destination pairs:

- o Shared bottleneck detection: As explained for instance in [I-D.ietf-alto-deployments], ALTO can be used to avoid excessive use of bottleneck links by recommending communication with endpoints not using that bottleneck. However, if multiple application transfers shall be scheduled, e.g., by a bandwidth calendaring application, it can be difficult to determine whether the resulting communication would hit a common bottleneck unless the applications are aware of links that would be used by multiple

transfers. This would benefit from an ALTO extension that can return shared parts of paths used by transfers between different endpoints (e.g., given by IP addresses).

- o Resilience improvement: Applications requiring high reliability can be interested in understanding if there is a risk of simultaneous failures on multiple path between application instances. For instance, transport networks can provide certain shared risk link group (SRLG) information that provides insight which links are at risk of simultaneous failures due to fate sharing. Applications requiring high reliability may then prefer the use of endpoints with disjoint paths.
- o Multi-homing and multi-pathing: If endpoints are multi-homed, i.e., if hosts have more than one IP address, it can be useful to know if there are multiple paths between these endpoints. For instance, applications could leverage such information to decide whether to explicitly enable Multipath TCP [RFC6824].

None of these use cases requires an application to understand routing protocol internals, such as the entries in Routing Information Base (RIB) of routers in the network. While it may be possible to derive such information from corresponding models, such as [I-D.clemm-i2rs-yang-network-topo], ALTO targets general-purpose applications that typically have no understanding of RIB data, different routing protocols, etc. Therefore, ALTO requires a more abstract solution to express topology details. An optional ALTO graph representation solution allows disclosure of such information to ALTO clients.

3.2. Topology Models

This document specifies two optional, abstract graph representation format for ALTO, which can reveal for instance coupling of links on a path. This section introduces the topology model. The formal specification follows in the next section.

There are two different graph representation formats that could be used to generalize the existing ALTO network and cost maps and reveal path coupling:

- o Path-vector format: This format describes the topology between given endpoints by an enumeration of the path traversed between the source and destination. The advantage of this format is that it can consider the outcome of network-internal routing policies (e.g., preferring paths other than the shortest path given by routing metrics). However, for a large network with many paths there is a large representation overhead.

- o Node-edge format: A graph encoding with nodes and edges can be very compact, depending on the average node degree, and it can also be much smaller than the full mesh of costs between endpoints that is used by ALTO cost maps. Yet, a downside is that a single graph may not convey network routing policies.

Since both formats have advantages and drawbacks, this document allows the use of both formats, and it is up to the ALTO service to decide whether to support it.

Both formats can reuse the PID concept that is used in ALTO to abstract topology details, as explained in [I-D.ietf-alto-protocol] and [I-D.ietf-alto-deployments].

4. Topology Encoding Formats for ALTO

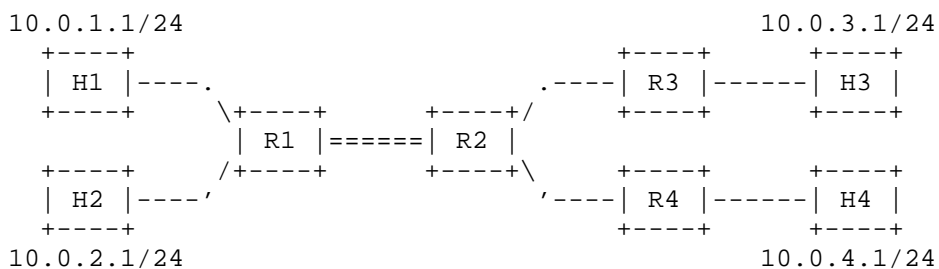
This section defines an OPTIONAL extension of ALTO for topology encoding in JSON format [RFC4627]. An ALTO service announces the support of these formats in the Information Resource Directory (IRD) [I-D.ietf-alto-protocol].

TODO: Currently, the formats are illustrated by examples only. The full formal specification is TBD.

4.1. Map Service Extension

The graph encoding uses the existing PID concept and ALTO map service [I-D.ietf-alto-protocol]. Basically, both in the path-vector and in the node-edge encoding, a PID is generalized as a node in the topology. If an ALTO server server uses either format, it MAY return PIDs without an IPv4 or IPv6 address mapping.

For instance, consider the following network topology. For the sake of simplicity, "H1", "H2", ... as well as "R1", "R2", ... are also used as PID names in the following examples.



R1, R2, ... are transit nodes in the topology. The objective of topology encoding formats is to represent their relations. Yet, the

addresses of those routers (e.g., interface or system IP addresses) may not matter to applications. Therefore, PIDs representing those nodes may not have a valid IP address mapping. In ALTO, it is up to an ALTO server to define what a PID represents. A PID can both refer to a single node (such as a router), a subnet, a whole network, etc.

A corresponding ALTO network map query according to [I-D.ietf-alto-protocol] would be:

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

For the given example, this would return:

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

```
{
  "meta": {
    ...
  },
  "network-map": {
    "H1": { "ipv4": [ "10.0.1.0/24" ] },
    "H2": { "ipv4": [ "10.0.2.0/24" ] },
    "H3": { "ipv4": [ "10.0.3.0/24" ] },
    "H4": { "ipv4": [ "10.0.4.0/24" ] },
    "R1": { }, "R2": { }, "R3": { }, "R4": { },
    "Default": {
      "ipv4": [ "0.0.0.0/0" ], "ipv6": [ "::/0" ]
    }
  }
}
```

R1, R2, R3, and R4 are transient PIDs on the path between endsystems. If their IP addresses do not matter, they can be omitted. If an identification of such nodes was required, one could introduce other identifiers. This is left for further study.

4.2. Path-Vector Format

This extension uses a new media type "application/alto-pathvector+json". An ALTO service MAY use the path-vector format as optional extension. In the path-vector format, an ALTO cost map consists of an ordered sequence of PIDs. Using the same query

mechanisms like the base ALTO protocol [I-D.ietf-alto-protocol], an ALTO query for path-vector information would be:

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

The corresponding response of an ALTO server would be:

```
HTTP/1.1 200 OK
Content-Length: TBA
Content-Type: application/alto-pathvector+json
```

```
{
  "meta": {
    ...
    "cost-type": {"cost-mode": "pathvector"}
  },
  "cost-map" : {
    "H1": {
      "H2": [ "R1" ],
      "H3": [ "R1", "R2", "R3" ]
      "H4": [ "R1", "R2", "R4" ]
    },
    "H2": { ... },
    "H3": { ... },
    "H4": { ... },
    "Default": { ... }
  }
}
```

In the most simple form, a path vector consists of an ordered sequence of PIDs from a source to a destination. It is also possible to extend the path vector format by integrating cost metrics in the vector. A corresponding format is left for further study in this document.

In general, the cost values between any PIDs can always be determined using the standard cost map of ALTO [I-D.ietf-alto-protocol]. As an example, the following query is also possible for the path-vector cost mode:

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

A corresponding response of an ALTO server could be, fully using the syntax and semantics of [I-D.ietf-alto-protocol]:

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

{
  "meta" : {
    ...
    "cost-type": { "cost-mode": "numerical",
                  "cost-metric": "routingcost"
                }
  },
  "cost-map": {
    "H1": { "R1": 1 },
    "H2": { "R1": 5 },
    "R1": { "H1": 1, "H2": 5, "R2": 9 },
    "R2": { "R1": 9, "R3": 4, "R4": 7 },
    ...
  }
}
```

In addition to the standard metric "routing cost", also other metrics for ALTO cost maps could be used, such as the ones described in [I-D.wu-alto-te-metrics].

4.3. Node-Edge Format

As an alternative representation, an ALTO service MAY also expose map data in a node-edge format. The node-edge format basically returns a list of edges between the PIDs given by the network map. The response SHOULD also include a list of the nodes, which is identical to the ALTO network map. Adding the list of nodes enables a client to process the full graph with a single query. This extension uses the new media type "application/alto-nodeedge+json". In the following a query for a node-edge map is shown:

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

In the given example, the response in node-edge format would be as follows:

HTTP/1.1 200 OK
 Content-Length: TBA
 Content-Type: application/alto-nodeedge+json

```
{
  "meta": {
    ...
    "cost-type": { "cost-mode": "nodeedge" }
  },
  "nodes": {
    "H1": { "ipv4": [ "10.0.1.0/24" ] },
    "H2": { "ipv4": [ "10.0.2.0/24" ] },
    "H3": { "ipv4": [ "10.0.3.0/24" ] },
    "H4": { "ipv4": [ "10.0.4.0/24" ] },
    "R1": { }, "R2": { }, "R3": { }, "R4": { },
    "Default": {
      "ipv4": [ "0.0.0.0/0" ], "ipv6": [ "::/0" ]
    }
  }
  "edges": [
    "E1": { "src": "H1", "dst": "R1",
      "type": "directed",
      "cost": [ { "cost-metric": "delay",
        "value": "3"
      }, {
        "cost-metric": "availbw",
        "value": "50"
      }, {
        "cost-metric": "risk-group",
        "value": [ "SLRG3" ]
      } ]
    },
    "E2": { "src": "R1", "dst": "R2",
      "type": "directed",
      "cost": [ { "cost-metric": "delay",
        "value": "3"
      }, {
        "cost-metric": "availbw",
        "value": "50"
      } ]
    },
    ...
  ]
}
```

The node-edge format re-uses the PID as identifiers for nodes. It requires a new set of identifiers for the edges. For those identifiers, the same syntax constraints like for PIDs are used.

Within one response, each edge must be uniquely identified by an edge identifier.

Edges can be characterized by the same attributes like ALTO cost maps, including one or more cost metrics. For instance, the cost metrics defined in [I-D.wu-alto-te-metrics] can be used for edges. The specification of further edge properties is for further study.

5. IANA Considerations

TBD

6. Security Considerations

TBD

7. Conclusion

TBD

8. References

8.1. Normative References

- [I-D.ietf-alto-protocol]
Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-27 (work in progress), March 2014.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.

8.2. Informative References

- [I-D.bernstein-alto-large-bandwidth-cases]
Bernstein, G. and Y. Lee, "Use Cases for High Bandwidth Query and Control of Core Networks", draft-bernstein-alto-large-bandwidth-cases-02 (work in progress), July 2012.
- [I-D.bernstein-alto-topo]
Bernstein, G., Yang, Y., and Y. Lee, "ALTO Topology Service: Uses Cases, Requirements, and Framework", draft-bernstein-alto-topo-00 (work in progress), October 2013.

- [I-D.clemm-i2rs-yang-network-topo]
Clemm, A., Ananthakrishnan, H., Medved, J., Tkacik, T.,
Varga, R., and N. Bahadur, "A YANG Data Model for Network
Topologies", draft-clemm-i2rs-yang-network-topo-00 (work
in progress), February 2014.
- [I-D.ietf-alto-deployments]
Stiemerling, M., Kiesel, S., Previdi, S., and M. Scharf,
"ALTO Deployment Considerations", draft-ietf-alto-
deployments-09 (work in progress), February 2014.
- [I-D.lee-alto-app-net-info-exchange]
Lee, Y., Dhody, D., Wu, Q., Bernstein, G., and T. Choi,
"ALTO Extensions to Support Application and Network
Resource Information Exchange for High Bandwidth
Applications in TE networks", draft-lee-alto-app-net-info-
exchange-04 (work in progress), October 2013.
- [I-D.wu-alto-te-metrics]
Wu, W., Yang, Y., Lee, Y., Dhody, D., and S. Randriamasy,
"ALTO Traffic Engineering Cost Metrics", draft-wu-alto-te-
metrics-03 (work in progress), June 2014.
- [I-D.yang-alto-topology]
Bernstein, G., Lee, Y., Roome, B., Scharf, M., and Y.
Yang, "ALTO Topology Extensions", draft-yang-alto-
topology-03 (work in progress), July 2014.
- [RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure,
"TCP Extensions for Multipath Operation with Multiple
Addresses", RFC 6824, January 2013.

Appendix A. Contributors

This document is an outcome of very valuable discussions with Y.
Richard Yang, Young Lee, and Greg M. Bernstein during IETF 89.

Author's Address

Michael Scharf
Alcatel-Lucent Bell Labs
Lorenzstrasse 10
Stuttgart 70435
Germany

Email: michael.scharf@alcatel-lucent.com

CDNI
Internet-Draft
Intended status: Informational
Expires: December 29, 2014

J. Seedorf
NEC
Y. Yang
Yale
J. Peterson
Neustar
June 27, 2014

CDNI Footprint and Capabilities Advertisement using ALTO
draft-seedorf-cdni-request-routing-alto-07

Abstract

Network Service Providers (NSPs) are currently considering to deploy Content Delivery Networks (CDNs) within their networks. As a consequence of this development, there is a need for interconnecting these local CDNs. The necessary interfaces for inter-connecting CDNs are currently being defined in the Content Delivery Networks Interconnection (CDNI) WG. This document focuses on the CDNI Footprint & Capabilities Advertisement interface (FCI). Specifically, this document specifies a new Application Layer Traffic Optimization (ALTO) service to facilitate Footprint & Capabilities Advertisement in a CDNI context.

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 December 29, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. ALTO within CDNI Request Routing	3
3. Assumptions and High-Level Design Considerations	4
3.1. General Assumptions and Considerations	4
3.2. Semantics for Footprint/Capabilities Advertisement	5
3.3. Advantages of using ALTO as the CDNI FCI protocol	7
3.4. Selection of a Downstream CDN with ALTO	7
4. CDNI FCI ALTO Service	8
4.1. Server Response Encoding	8
4.1.1. CDNI FCI Map	8
4.1.2. Meta Information	8
4.1.3. Data Information	8
4.2. Protocol Errors	8
4.3. Example	9
5. Useful ALTO extensions for CDNI Request Routing	10
6. Security Considerations	11
7. Acknowledgements	11
8. References	11
8.1. Normative References	11
8.2. Informative References	12
Authors' Addresses	13

1. Introduction

Many Network Service Providers (NSPs) are currently considering or have already started to deploy Content Delivery Networks (CDNs) within their networks. As a consequence of this development, there is a need for interconnecting these local CDNs. Content Delivery Networks Interconnection (CDNI) has the goal of standardizing protocols to enable such interconnection of CDNs [RFC6707].

The CDNI problem statement [RFC6707] envisions four interfaces to be standardized within the IETF for CDN interconnection:

- o CDNI Request Routing Interface

- o CDNI Metadata Interface
- o CDNI Logging Interface
- o CDNI Control Interface

This document focuses solely on the CDNI Request Routing Interface, which can be further divided into two interfaces (see [RFC6707] for a detailed description): the CDNI Request Routing Redirection interface (RI), and the CDNI Footprint & Capabilities Advertisement interface (FCI). This document specifies a new Application Layer Traffic Optimization (ALTO) [I-D.ietf-alto-protocol] service called 'CDNI Footprint & Capabilities Advertisement Service'. This service is used to transport a CDNI FCI JSON objects, which are defined in a separate document [I-D.ma-cdni-capabilities].

Throughout this document, we use the terminology for CDNI defined in [I-D.ietf-cdni-problem-statement].

2. ALTO within CDNI Request Routing

The main purpose of the CDNI Request Routing Interface is described in [RFC6707] as follows: "The CDNI Request Routing interface enables a Request Routing function in an Upstream CDN to query a Request Routing function in a Downstream CDN to determine if the Downstream CDN is able (and willing) to accept the delegated Content Request. It also allows the Downstream CDN to control what should be returned to the User Agent in the redirection message by the upstream Request Routing function." On a high level, the scope of the CDNI Request Routing Interface therefore contains two main tasks:

- o A) Determining if the downstream CDN is willing to accept a delegated content request
- o B) Redirecting the content request coming from an upstream CDN to the proper entry point or entity in the downstream CDN

More precisely, in [I-D.ietf-cdni-framework] the request routing interface is broadly divided into two functionalities:

- o 1) the asynchronous advertisement of footprint and capabilities by a dCDN that allows a uCDN to decide whether to redirect particular user requests to that dCDN (the CDNI FCI)
- o 2) the synchronous operation of actually redirecting a user request (the CDNI RI)

Application Layer Traffic Optimization (ALTO) [I-D.ietf-alto-protocol] is an approach for guiding the resource provider selection process in distributed applications that can choose among several candidate resources providers to retrieve a given resource. By conveying network layer (topology) information, an ALTO server can provide important information to "guide" the resource provider selection process in distributed applications. Usually, it is assumed that an ALTO server conveys information these applications cannot measure themselves [RFC5693].

Originally, ALTO was motivated by the huge amount of cross-ISP traffic generated by P2P applications [RFC5693]. Recently, however, ALTO is also being considered for improving the request routing in CDNs [I-D.jenkins-alto-cdn-use-cases]. In this context, it has also been proposed to use ALTO for selecting an entry-point in a downstream NSP's network (see section 3.4 "CDN delivering Over-The-Top of a NSP's network" in [I-D.jenkins-alto-cdn-use-cases]). Also, the CDNI problem statement explicitly mentions ALTO as a candidate protocol for "algorithms for selection of CDN or Surrogate by Request-Routing systems" [I-D.ietf-cdni-problem-statement].

3. Assumptions and High-Level Design Considerations

In this section we list some assumptions and design issues to be considered when using ALTO for the CDNI Footprint and Capabilities Advertisement interface.

3.1. General Assumptions and Considerations

Below we list some general assumptions and considerations:

- o As explicitly being out-of-scope for CDNI [I-D.ietf-cdni-problem-statement], it is assumed that ingestion of content or acquiring content across CDNs is not part of request routing as considered within CDNI standardization work. The focus of using ALTO (as considered in this document) is hence on request routing only, assuming that the content (desired by the end user) is available in the downstream CDN (or can be acquired by the downstream CDN by some means).
- o Federation Model: "Footprint and Capabilities Advertisement" and in general CDN request routing depends on the federation model among the CDN providers. Designing a suitable solution thus depends on whether a solution is needed for different settings, where CDNs consist of both NSP CDNs (serving individual ASes) and general, traditional CDNs (such as Akamai). We assume that CDNI is not designed for a setting where only NSP CDNs each serve a single AS only.

- o In this document, it is assumed that the upstream CDN (uCDN) makes the decision on selecting a downstream CDN, based on information that each downstream CDN has made available to the upstream CDN. Further, we assume that in principle more than one dCDN may be suitable for a given end-user request (i.e. different dCDNs may claim "overlapping" footprints). The uCDN hence potentially has to select among several candidate downstream CDNs for a given end user request.
- o It is not clear what kind(s) of business, contract, and operational relationships two peering CDNs may form. For the Internet, we see provider-customer and peering as two main relations; providers may use different charging models (e.g., 95-percentile, total volume) and may provide different SLAs. Given such unknown characteristics of CDN peering business agreements, we should design the protocol to support as much diverse potential business and operational models as possible.

3.2. Semantics for Footprint/Capabilities Advertisement

The CDNI document on "Footprint and Capabilities Semantics" [I-D.ietf-cdni-footprint-capabilities-semantics] defines the semantics for the CDNI FCI. It thus provides guidance on what Footprint and Capabilities mean in a CDNI context and how a protocol solution should in principle look like. Here we briefly summarize the key points of the semantics of Footprint and Capabilities (for a detailed discussion, the reader is referred to [I-D.ietf-cdni-footprint-capabilities-semantics]):

- o Often, footprint and capabilities are tied together and cannot be interpreted independently from each other. In such cases, i.e. where capabilities must be expressed on a per footprint basis, it may be beneficial to combine footprint and capabilities advertisement.
- o Given that a large part of Footprint and Capabilities Advertisement will actually happen in contractual agreements, the semantics of CDNI Footprint and Capabilities advertisement refer to answering the following question: what exactly still needs to be advertised by the CDNI FCI? For instance, updates about temporal failures of part of a footprint can be useful information to convey via the CDNI request routing interface. Such information would provide updates on information previously agreed in contracts between the participating CDNs. In other words, the CDNI FCI is a means for a dCDN to provide changes/updates regarding a footprint and/or capabilities it has prior agreed to serve in a contract with a uCDN.

- o It seems clear that "coverage/reachability" types of footprint must be supported within CDNI. The following such types of footprint are mandatory and must be supported by the CDNI FCI:

- * List of ISO Country Codes
- * List of AS numbers
- * Set of IP-prefixes

A 'set of IP-prefixes' must be able to contain full IP addresses, i.e., a /32 for IPv4 and a /128 for IPv6, and also IP prefixes with an arbitrary prefix length. There must also be support for multiple IP address versions, i.e., IPv4 and IPv6, in such a footprint.

- o For all of these mandatory-to-implement footprint types, footprints can be viewed as constraints for delegating requests to a dCDN: A dCDN footprint advertisement tells the uCDN the limitations for delegating a request to the dCDN. For IP prefixes or ASN(s), the footprint signals to the uCDN that it should consider the dCDN a candidate only if the IP address of the request routing source falls within the prefix set (or ASN, respectively). The CDNI specifications do not define how a given uCDN determines what address ranges are in a particular ASN. Similarly, for country codes a uCDN should only consider the dCDN a candidate if it covers the country of the request routing source. The CDNI specifications do not define how a given uCDN determines the country of the request routing source. Multiple footprint constraints are additive, i.e. the advertisement of different types of footprint narrows the dCDN candidacy cumulatively.
- o The following capabilities seem useful as 'base' capabilities, i.e. ones that are needed in any case and therefore constitute mandatory capabilities to be supported by the CDNI FCI:
 - * Delivery Protocol (e.g., HTTP vs. RTMP)
 - * Acquisition Protocol (for acquiring content from a uCDN)
 - * Redirection Mode (e.g., DNS Redirection vs. HTTP Redirection as discussed in [I-D.ietf-cdni-framework])
 - * Capabilities related to CDNI Logging (e.g., supported logging mechanisms)

- * Capabilities related to CDNI Metadata (e.g., authorization algorithms or support for proprietary vendor metadata)

3.3. Advantages of using ALTO as the CDNI FCI protocol

The following reasons make ALTO a suitable candidate protocol for downstream CDN selection as part of CDNI request routing and in particular for an FCI protocol:

- o CDN request routing is done at the application layer. ALTO is a protocol specifically designed to improve application layer traffic (and application layer connections among hosts on the Internet) by providing additional information to applications that these applications could not easily retrieve themselves. For CDNI, this is exactly the case: a uCDN wants to improve application layer CDN request routing by using dedicated information (provided by a dCDN) that the uCDN could not easily obtain otherwise.
- o The semantics of an ALTO network map are an exact match for the needed information to convey a footprint by a downstream CDN, in particular if such a footprint is being expressed by IP-prefix ranges.
- o Security: ALTO maps can be signed and hence provide inherent integrity protection (see Section 6)
- o RESTful-Design: The ALTO protocol has undergone extensive revisions in order to provide a RESTful design regarding the client-server interaction specified by the protocol. A CDNI FCI interface based on ALTO would inherit this RESTful design.
- o Error-handling: The ALTO protocol has undergone extensive revisions in order to provide sophisticated error-handling, in particular regarding unexpected cases. A CDNI FCI interface based on ALTO would inherit this thought-through and mature error-handling.
- o Filtered network map: The ALTO Map Filtering Service (see [I-D.ietf-alto-protocol] for details) would allow a uCDN to query only for parts of an ALTO map.

3.4. Selection of a Downstream CDN with ALTO

Under the considerations stated in Section 3, ALTO can help the upstream CDN provider to select a proper downstream CDN provider for a given end user request as follows: Each downstream CDN provider

hosts an ALTO server which provides ALTO services which convey CDNI FCI information to an ALTO client at the upstream CDN provider.

4. CDNI FCI ALTO Service

The ALTO protocol is based on an ALTO Information Service Framework which consists of several services, where all ALTO services are 'provided through a common transport protocol, messaging structure and encoding, and transaction model' [I-D.ietf-alto-protocol]. The ALTO protocol specification [I-D.ietf-alto-protocol] defines several such services, e.g. the ALTO map service.

This document defines a new ALTO Service called 'CDNI Footprint & Capabilities Advertisement Service' which conveys JSON objects of media type 'application/alto-fcimap+json'. This media type and JSON object format is defined in [I-D.ma-cdni-capabilities]; this document specifies how to transport such JSON objects via the ALTO protocol with the ALTO 'CDNI Footprint & Capabilities Advertisement Service'.

4.1. Server Response Encoding

4.1.1. CDNI FCI Map

The media type of the CDNI FCI Map is 'application/alto-cdni-fcimap+json'. The HTTP Method, Accept Input Parameters, Capabilities, Uses, and Response of the CDNI FCI Map are specified in [I-D.ma-cdni-capabilities].

4.1.2. Meta Information

The 'meta' field of a FCIMapData response MUST include 'vtag', which is an ALTO Version Tag of the retrieved FCIMapData according to [I-D.ietf-alto-protocol] (Section 10.3.). It thus contains a 'resource-id' attribute, and a 'tag' is an identifier string.

4.1.3. Data Information

The data component of a CDNI FCI Map resource is named 'fcimap' which is a JSON object of type FCIMapData. This JSON object of type FCIMapData is derived from ResponseEntityBase as specified in the ALTO protocol [I-D.ietf-alto-protocol] (Section 8.4.) and specified in [I-D.ma-cdni-capabilities].

4.2. Protocol Errors

Protocol errors are handled as specified in the ALTO protocol [I-D.ietf-alto-protocol] (Section 8.5.).

4.3. Example

The following example shows an CDNI FCI Map as in [I-D.ma-cdni-capabilities], however with meta-information as defined in Section 4.1.2 of this document.

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

HTTP/1.1 200 OK
Content-Length: 439
Content-Type: application/alto-fcimap+json
{
  "meta" : {
    "vtag": {
      "resource-id": "my-default-fcimap",
      "tag": "da65eca2eb7a10ce8b059740b0b2e3f8eb1d4785"
    }
  },
  "fcimap": [
    { "name": "delivery_protocol",
      "values": [
        "HTTP",
        "RTSP",
        "MMS"
      ]
    },
    { "name": "delivery_protocol",
      "values": [
        "RTMP",
        "HTTPS"
      ],
      "footprint": [
        { "type": "IPv4CIDR",
          "values": [
            "10.1.0.0/16",
            "10.10.10.0/24"
          ]
        }
      ]
    }
  ]
}
```

5. Useful ALTO extensions for CDNI Request Routing

It is envisioned that yet-to-be-defined ALTO extensions will be standardized that make the ALTO protocol more suitable and useful for applications other than the originally considered P2P use case [I-D.marocco-alto-next]. Some of these extensions to the ALTO protocol would be useful for ALTO to be used as a protocol within CDNI request routing, and in particular within the "Footprint and Capabilities Advertisement" part of the CDNI request routing interface.

The following proposed extensions to ALTO would be beneficial to facilitate CDNI request routing with ALTO as outlined in Section 3.4:

- o Server-initiated Notifications and Incremental Updates: In case the footprint or the capabilities of a downstream CDN change abruptly (i.e. unexpectedly from the perspective of an upstream CDN), server initiated notifications would enable a dCDN to directly inform an upstream CDN about such changes. Consider the case where - due to failure - part of the footprint of the dCDN is not functioning, i.e. the CDN cannot serve content to such clients with reasonable QoS. Without server-initiated notifications, the uCDN might still use a very recent network and cost map from dCDN, and therefore redirect request to dCDN which it cannot serve. Similarly, the possibility for incremental updates would enable efficient conveyance of the aforementioned (or similar) status changes by the dCDN to the uCDN. A proposal for server-initiated ALTO updates can be found in [I-D.marocco-alto-ws]. A discussion of incremental ALTO updates can be found in [I-D.schwan-alto-incr-updates].
- o Content Availability on Hosts: A dCDN might want to express CDN capabilities in terms of certain content types (e.g. codecs/formats, or content from certain content providers). A new endpoint property for ALTO that would be able to express such "content availability" would enable a dCDN to make available such information to an upstream CDN. This would enable a uCDN to determine if a given dCDN actually has the capabilities for a given request with respect to the type of content requested.
- o Resource Availability on Hosts or Links: The capabilities on links (e.g. maximum bandwidth) or caches (e.g. average load) might be useful information for an upstream CDN for optimized downstream CDN selection. For instance, if a uCDN receives a streaming request for content with a certain bitrate, it needs to know if it is likely that a dCDN can fulfill such stringent application-level requirements (i.e. can be expected to have enough consistent bandwidth) before it redirects the request. In general, if ALTO

could convey such information via new endpoint properties, it would enable more sophisticated means for downstream CDN selection with ALTO.

6. Security Considerations

One important security consideration is the proper authentication of advertisement information provided by a downstream CDN. The ALTO protocol provides a specification for a signature of ALTO information (see 8.2.2. of [I-D.ietf-alto-protocol]). ALTO thus provides a proper means for protecting the integrity of FCI information.

More Security Considerations will be discussed in a future version of this document.

7. Acknowledgements

The authors would like to thank Kevin Ma, Daryl Malas, and Matt Caulfield for their timely reviews and invaluable comments.

Jan Seedorf is partially supported by the GreenICN project (GreenICN: Architecture and Applications of Green Information Centric Networking), a research project supported jointly by the European Commission under its 7th Framework Program (contract no. 608518) and the National Institute of Information and Communications Technology (NICT) in Japan (contract no. 167). 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 GreenICN project, the European Commission, or NICT.

8. References

8.1. Normative References

- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.
- [RFC6707] Niven-Jenkins, B., Le Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", RFC 6707, September 2012.
- [RFC6770] Bertrand, G., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", RFC 6770, November 2012.

8.2. Informative References

[I-D.peterson-CDNI-strawman]

Peterson, L. and J. Hartman, "Content Distribution Network Interconnection (CDNI) Problem Statement", draft-peterson-CDNI-strawman-01 (work in progress), May 2011.

[I-D.ietf-cdni-problem-statement]

Niven-Jenkins, B., Faucheur, F., and N. Bitar, "Content Distribution Network Interconnection (CDNI) Problem Statement", draft-ietf-cdni-problem-statement-08 (work in progress), June 2012.

[I-D.marocco-alto-next]

Marocco, E. and V. Gurbani, "Extending the Application-Layer Traffic Optimization (ALTO) Protocol", draft-marocco-alto-next-00 (work in progress), January 2012.

[I-D.ietf-alto-protocol]

Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-27 (work in progress), March 2014.

[I-D.ietf-cdni-requirements]

Leung, K. and Y. Lee, "Content Distribution Network Interconnection (CDNI) Requirements", draft-ietf-cdni-requirements-17 (work in progress), January 2014.

[I-D.ietf-cdni-use-cases]

Bertrand, G., Emile, S., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", draft-ietf-cdni-use-cases-10 (work in progress), August 2012.

[I-D.marocco-alto-ws]

Marocco, E. and J. Seedorf, "WebSocket-based server-to-client notifications for the Application-Layer Traffic Optimization (ALTO) Protocol", draft-marocco-alto-ws-02 (work in progress), February 2014.

[I-D.schwan-alto-incr-updates]

Schwan, N. and B. Roome, "ALTO Incremental Updates", draft-schwan-alto-incr-updates-02 (work in progress), July 2012.

- [I-D.jenkins-alto-cdn-use-cases]
Niven-Jenkins, B., Watson, G., Bitar, N., Medved, J., and S. Previdi, "Use Cases for ALTO within CDNs", draft-jenkins-alto-cdn-use-cases-03 (work in progress), June 2012.
- [I-D.ma-cdni-capabilities]
Ma, K. and J. Seedorf, "CDNI Footprint & Capabilities Advertisement Interface", draft-ma-cdni-capabilities-05 (work in progress), June 2014.
- [I-D.ietf-cdni-framework]
Peterson, L., Davie, B., and R. Brandenburg, "Framework for CDN Interconnection", draft-ietf-cdni-framework-14 (work in progress), June 2014.
- [I-D.liu-cdni-cost]
Liu, H., "A Cost Perspective on Using Multiple CDNs", draft-liu-cdni-cost-00 (work in progress), October 2011.
- [I-D.spp-cdni-rr-foot-cap-semantics]
Seedorf, J., Peterson, J., Previdi, S., Brandenburg, R., and K. Ma, "CDNI Request Routing: Footprint and Capabilities Semantics", draft-spp-cdni-rr-foot-cap-semantics-04 (work in progress), February 2013.
- [I-D.ietf-cdni-metadata]
Niven-Jenkins, B., Murray, R., Watson, G., Caulfield, M., Leung, K., and K. Ma, "CDN Interconnect Metadata", draft-ietf-cdni-metadata-06 (work in progress), February 2014.
- [I-D.ietf-cdni-logging]
Faucheur, F., Bertrand, G., Oprescu, I., and R. Peterkofsky, "CDNI Logging Interface", draft-ietf-cdni-logging-11 (work in progress), March 2014.
- [I-D.ietf-cdni-footprint-capabilities-semantics]
Seedorf, J., Peterson, J., Previdi, S., Brandenburg, R., and K. Ma, "CDNI Request Routing: Footprint and Capabilities Semantics", draft-ietf-cdni-footprint-capabilities-semantics-02 (work in progress), February 2014.

Authors' Addresses

Jan Seedorf
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 221
Email: jan.seedorf@neclab.eu
URI: <http://www.neclab.eu>

Y.R. Yang
Yale University
51 Prospect Street
New Haven 06511
USA

Email: yry@cs.yale.edu
URI: <http://www.cs.yale.edu/~yry/>

Jon Peterson
NeuStar
1800 Sutter St Suite 570
Concord CA 94520
USA

Email: jon.peterson@neustar.biz

ALTO Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2014

Q. Wu
Huawei
Y. Yang
Yale University
Y. Lee
D. Dhody
Huawei
S. Randriamasy
Alcatel-Lucent
June 27, 2014

ALTO Traffic Engineering Cost Metrics
draft-wu-alto-te-metrics-03

Abstract

Cost Metric is a basic concept in Application-Layer Traffic Optimization (ALTO). It is used in both the Cost Map Service and the Endpoint Cost Service. Future extensions to ALTO may also use Cost Metric.

Different applications may benefit from different Cost Metrics. For example, a Resource Consumer may prefer Resource Providers that have low delay to the Resource Consumer. However the base ALTO protocol [ALTO] has defined only a single cost metric, i.e., the generic "routingcost" metric (Sec. 14.2 of ALTO base specification [ALTO]).

In this document, we define eleven Cost Metrics, derived from OSPF-TE and ISIS-TE, to measure network delay, jitter, packet loss, hop count, and bandwidth. The metrics defined in this document provide a relatively comprehensive set of Cost Metrics for ALTO focusing on traffic engineering (TE). Additional Cost Metrics such as financial cost metrics may be defined in other documents.

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 December 29, 2014.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Data sources, computation of defined cost metrics	4
2.1. Data sources	4
2.2. Computation of metrics	4
3. Metric: Delay	5
4. Metric: Delay Jitter	6
5. Metric: Packet Loss	8
6. Metric: Hop Count	10
7. Metric: Bandwidth	12
8. Metric: Maximum Bandwidth	13
9. Metric: Maximum Reserved Bandwidth	14
10. Metric: Unavailable Reserved Bandwidth	16
11. Metric: Residue Bandwidth	19
12. Metric: Available Bandwidth	20
13. Metric: Utilized Bandwidth	22
14. Security Considerations	24
15. IANA Considerations	25
16. References	25
16.1. Normative References	25
16.2. Informative References	26
Authors' Addresses	26

1. Introduction

Cost Metric is a basic concept in Application-Layer Traffic Optimization (ALTO). It is used in both the Cost Map Service and the Endpoint Cost Service. In particular, applications may benefit from knowing network performance measured on several Cost Metrics. For example, a more delay sensitive application may focus on latency, and a more bandwidth-sensitive application may focus on available bandwidth.

The objective of this document is to define eleven cost metrics, listed in Table 1, to support the aforementioned applications. Hence, this document extends the base ALTO protocol [ALTO], which defines only a single cost metric, i.e., the generic "routingcost" metric (Sec. 14.2 of ALTO base specification [ALTO]).

Namespace	Property	Reference
	delay	[RFCxxxx], Section 3
	jitter	[RFCxxxx], Section 4
	pktloss	[RFCxxxx], Section 5
	hopcount	[RFCxxxx], Section 6
	bandwidth	[RFCxxxx], Section 7
	maxbw	[RFCxxxx], Section 8
	maxresbw	[RFCxxxx], Section 9
	unresdbw	[RFCxxxx], Section 10
	residbw	[RFCxxxx], Section 11
	availbw	[RFCxxxx], Section 12
	utilbw	[RFCxxxx], Section 13

Table 1.

An ALTO server may provide a subset of the cost metrics defined in this document. When an ALTO server supports a cost metric defined in this document, the server SHOULD announce the metric in its IRD.

The definitions of a set of cost metrics can allow us to extend the ALTO base protocol (e.g., allowing output and constraints use different cost metrics), but such extensions are not in the scope of this document.

One challenge in defining the metrics is that performance metrics often depend on configuration parameters. For example, the value of packet loss rate depends on the measurement interval and varies over time. To handle this issue, ALTO server may collect data on time periods covering the past, present or only collect data on present time.

Following the ALTO base protocol, this document uses JSON to specify the value type of each defined metric. See [RFC4627] for JSON data type specification.

2. Data sources, computation of defined cost metrics

The cost metrics defined in this document are similar, in that they may use similar data sources and have similar issues in their calculation. Hence, instead of specifying such issues for each metric individually, we specify the common issue in this section.

2.1. Data sources

An ALTO server needs data sources to compute the cost metrics defined in this document. This document does not define the exact data sources. For example, the ALTO server may use log servers or the OAM system as its data source [ALTO-DEPLOYMENT]. In particular, the cost metrics defined in this document can be computed using routing systems as the data sources. Mechanisms defined in [RFC3630], [RFC3784], [OSPF-TE], [ISIS-TE], [BGP-LS] and [BGP-PM] that allow an ALTO Server to retrieve and derive the necessary information to compute the metrics that we define in this document.

2.2. Computation of metrics

An ALTO server process measurements from data sources to compute exposed metrics. It may need performance data processing tasks such as aggregating the results across multiple systems, removing outliers, and creating additional statistics.

One specific challenge in defining the metrics in this document is that these performance metrics depend on some configuration parameters. For example, the value of packet loss rate depends on the measurement interval and varies over time. If the ALTO server uses aforementioned routing protocol based mechanisms as data sources, then the measurement interval may be preconfigured by the routing protocol. For example, Section 5 of [ISIS-TE] defines a default measurement interval of 30 seconds. This document uses the term Measurement Interval to refer to the measurement interval used by the data sources. The Measurement Interval(s) of the data sources can be different from the interval that this document defines the metric. Hence, an ALTO server needs to resolve the mismatch, when it happens. [TODO: Need more specification.]

Another issue of converting from data source measurements to ALTO exposed metric values is that the measurement results that the ALTO Server retrieves may be defined for only links, and hence, the server will need to compose the link metrics to obtain path metrics used in

services such as the Cost Map Service. In this definition, we define the metrics to be independent of link or path, considering that future ALTO extensions may define link-based services, and hence the defined metrics should still be usable.

3. Metric: Delay

Cost Metric name:

Delay

Cost Metric string:

US-ASCII string 'delay'

Metric Description:

To specify spatial and temporal aggregated delay between the specified source and destination or the time that the packet spends to travel from source to destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit:

The unit is microsecond.

Metric Value Type:

A single 'JSONNumber' type value containing a non-negative integer component that may be followed by an exponent part.

Example 1: Delay value on source-destination endpoint pairs

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

```
{
  "cost-type": { "cost-mode" : "numerical",
                 "cost-metric" : "delay" },
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta" : {
    "cost-type": { "cost-mode" : "numerical",
                   "cost-metric" : "delay" }
  },
  "endpoint-cost-map" : {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89" : 10,
      "ipv4:198.51.100.34" : 20,
      "ipv4:203.0.113.45" : 30,
    }
  }
}
```

4. Metric: Delay Jitter

Cost Metric name:

Delay jitter

Cost Metric string:

US-ASCII string 'jitter'

Metric Description:

To specify spatial and temporal aggregated jitter (latency variation) over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit:

The unit is microsecond.

Metric Value Type:

A single 'JSONNumber' type value containing an integer component that may be followed by exponent part.

Example 2: Delayjitter value on source-destination endpoint pairs

POST /endpointcost/lookup HTTP/1.1

Host: alto.example.com

Content-Length: TBA

Content-Type: application/alto-endpointcostparams+json

Accept: application/alto-endpointcost+json,application/alto-error+json

```
{
  "cost-type": { "cost-mode" : "numerical",
                 "cost-metric" : "jitter" },
  "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: TBA

Content-Type: application/alto-endpointcost+json

```
{
  "meta": {
    "cost type": {
      "cost-mode": "numerical",
      "cost-metric": "jitter"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89" : 0
      "ipv4:198.51.100.34" : 1
      "ipv4:203.0.113.45" : 5
    }
  }
}
```

5. Metric: Packet Loss

Cost Metric name:

Packet loss

Cost Metric string:

US-ASCII string 'pktloss'

Metric Description:

To specify spatial and temporal aggregated packet loss over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit:

The unit is percentile.

Metric Value Type:

A single number value containing an integer component that may be followed by a fraction part and/or an exponent part.

Example 3: pktloss value on source-destination endpoint pairs

POST /endpointcost/lookup HTTP/1.1

Host: alto.example.com

Content-Length: TBA

Content-Type: application/alto-endpointcostparams+json

Accept: application/alto-endpointcost+json,application/alto-error+json

```
{
  "cost-type": { "cost-mode" : "numerical",
                 "cost-metric" : "pktloss" },
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost type": {
      "cost-mode": "numerical",
      "cost-metric": "pktloss"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89" : 0,
      "ipv4:198.51.100.34": 1,
      "ipv4:203.0.113.45" : 2,
    }
  }
}
```

6. Metric: Hop Count

The metric hopcount is mentioned in [ALTO] as an example. This section further clarifies its properties.

Cost Metric name:

Hop count

Cost Metric string:

US-ASCII string 'hopcount'

Metric Description:

To specify the number of hops in the path between the source endpoint and the destination endpoint.

[Editor Note: Need to specify which level (AS, IP perhaps), details TBD for multiple-layer aspect.]

Metric Unit:

The unit is integer number.

Metric Value Type:

A single 'JSONNumber' type value containing an integer component.

Example 4: hopcount value on source-destination endpoint pairs

POST /endpointcost/lookup HTTP/1.1

Host: alto.example.com

Content-Length: TBA

Content-Type: application/alto-endpointcostparams+json

Accept: application/alto-endpointcost+json,application/alto-error+json

```
{
  "cost-type": { "cost-mode" : "numerical",
                 "cost-metric" : "hopcount" },
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost type": {
      "cost-mode": "numerical",
      "cost-metric": "hopcount"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89" : 5,
      "ipv4:198.51.100.34": 3,
      "ipv4:203.0.113.45" : 2,
    }
  }
}
```

7. Metric: Bandwidth

Cost Metric name:

Bandwidth

Cost Metric string:

US-ASCII string 'bandwidth'

Metric Description:

To specify spatial and temporal aggregated bandwidth over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endhost to endhost); and the temporal unit is specified as the measurement interval in the query context.

This is just a definition of a class of cost metric 'bandwidth'. The use of this cost metric is always in conjunction with what it represents, which could be Max Bandwidth (maxbw), Residual Bandwidth (residuebw) etc.

Metric Unit:

The units are bytes per second.

Metric Value Type:

A single 'JSONNumber' type value containing an integer component , which may be followed by a fraction part and/or an exponent part.

8. Metric: Maximum Bandwidth**Cost Metric name:**

Maximum Bandwidth

Cost Metric string:

US-ASCII string 'maxbw'

Metric Description:

To specify spatial and temporal maximum bandwidth over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endhost to endhost); and the temporal unit is specified as the measurement interval in the context interval.

Metric Unit and Metric Value Type:

See definition for the Bandwidth Cost Metric.

Example 5: maxbw value on source-destination endpoint pairs

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

```
{
  "cost-type": { "cost-mode": "numerical",
  "cost-metric": "maxbw"},
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "maxbw"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2": {
      "ipv4:192.0.2.89": 0,
      "ipv4:198.51.100.34" : 2000,
      "ipv4:203.0.113.45": 5000,
    }
  }
}
```

9. Metric: Maximum Reserved Bandwidth

Cost Metric name:

Maximum Reserved Bandwidth

Cost Metric string:

US-ASCII string 'maxresbw'

Metric Description:

To specify spatial and temporal maximum reserved bandwidth over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit and Value Type:

See definition of the Bandwidth Cost Metric.

Example 6: maxresbw value on source-destination endpoint pairs

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

```
{
  "cost-type" { "cost-mode": "numerical",
    "cost-metric": "maxresbw"},
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "maxresbw"
    }
  },
  " endpoint-cost-map": {
    "ipv4:192.0.2.2" {
      "ipv4:192.0.2.89" : 0,
      "ipv4:198.51.100.34": 2000,
      "ipv4:203.0.113.45": 5000,
    }
  }
}
```

10. Metric: Unavailable Reserved Bandwidth

Cost Metric name:

Unavailable Reserved Bandwidth

Cost Metric string:

US-ASCII string 'unresbw'

Metric Description:

To specify spatial and temporal unavailable reserved bandwidth over the specified source and destination. The values correspond to the bandwidth that can be reserved with a setup priority of 0 through 7. Therefore this metric is encoded as an array of 8 values. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit and Value Type:

See definition for the bandwidth Cost Metric.

Example 7: unresbw value on source-destination endpoint pairs
In this example, the Collection method specifies that the 'unresbw' values are defined as the 'unavailable bandwidth' specified in section 2.5.8 of RFC3630: 8 unavailable bandwidth value are reported in the same OSPF message using the same TLV. Each value is corresponding to the bandwidth that can be reserved with a setup priority of 0 through 7.

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

```
{
  "cost-type" { "cost-mode": "numerical",
  "cost-metric": "unresbw[1,8]" },
  "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: TBA
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numerical",
      "cost-metric": "unresbw[1,8]"
    }
  },
  "endpoint-cost-map" {
    "ipv4:192.0.2.2" {
      "ipv4:192.0.2.89" : [0,0,0,0,0,0,0,0],
      "ipv4:198.51.100.34": [0,0,0,0,0,0,0,2000],
      "ipv4:203.0.113.45": [0,0,0,0,0,0,0,5000],
    }
  }
}
```

11. Metric: Residue Bandwidth

Cost Metric name:

Residue Bandwidth

Cost Metric string:

US-ASCII string 'residbw'

Metric Description:

To specify spatial and temporal residual bandwidth over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit and Value Type:

See definition of the general Bandwidth.

Example 8: residubw value on source-destination endpoint pairs

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

```
{
  "cost-type": { "cost-mode": "numerical",
                 "cost-metric": "residubw"},
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost-type" {
      "cost-mode": "numerical",
      "cost-metric": "residubw"
    }
  },
  "endpoint-cost-map" {
    "ipv4:192.0.2.2" {
      "ipv4:192.0.2.89" : 0,
      "ipv4:198.51.100.34": 2000,
      "ipv4:203.0.113.45": 5000,
    }
  }
}
```

12. Metric: Available Bandwidth

Cost Metric name:

Available Bandwidth

Cost Metric string:

US-ASCII string 'availbw'

Metric Description:

To specify spatial and temporal available bandwidth over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit and Value Type:

See definition of the general Bandwidth.

Example 9: availbw value on source-destination endpoint pairs

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

```
{
  "cost-type": { "cost-mode": "numeric",
                 "cost-metric": "availbw" },
  "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: TBA
Content-Type: application/alto-endpointcost+json
```

```
{
  "meta": {
    "cost-type": {
      "cost-mode": "numeric",
      "cost-metric": "availbw"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2" : {
      "ipv4:192.0.2.89" : [6,5,7,8,4,10,7,6],
      "ipv4:198.51.100.34" : [7,4,6,8,5,9,6,7],
      "ipv4:203.0.113.45" : [7,6,8,5,7,9,6,8],
    }
  }
}
```

13. Metric: Utilized Bandwidth

Cost Metric name:

Utilized Bandwidth

Cost Metric string:

US-ASCII string 'utilbw'

Metric Description:

To specify spatial and temporal utilized bandwidth over the specified source and destination. The spatial aggregation unit is specified in the query context (e.g., PID to PID, or endpoint to endpoint); and the temporal unit is specified as the measurement interval in the query context.

Metric Unit and Value Type:

See definition of the general Bandwidth.

Example 10: utilbw value on source-destination endpoint pairs

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

```
{
  "cost-type": { "cost-mode" : "numerical",
                 "cost-metric" : "utilbw" },
  "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: TBA
Content-Type: application/alto-endpointcost+json
{
  "meta": {
    "cost type": {
      "cost-mode": "numerical",
      "cost-metric": "utilbw"
    }
  },
  "endpoint-cost-map": {
    "ipv4:192.0.2.2" {
      "ipv4:192.0.2.89" : 0,
      "ipv4:198.51.100.34" : 2000,
      "ipv4:203.0.113.45" : 5000,
    }
  }
}
```

14. Security Considerations

The properties defined in this document present no security considerations beyond those in Section 15 of the base ALTO specification [ALTO].

However concerns addressed in Sections "15.1 Authenticity and Integrity of ALTO Information", "15.2 Potential Undesirable Guidance

from Authenticated ALTO Information" and "15.3 Confidentiality of ALTO Information" remain of utmost importance. Indeed, TE performance is a highly sensitive ISP information and sharing TE metric values in numerical mode requires full mutual confidence between the entities managing the ALTO Server and Client. Numerical TE performance information will most likely be distributed by ALTO Servers to Clients under strict and formal mutual trust agreements. One the other hand, ALTO Clients must be cognizant on the risks attached to such information that they would have acquired outside formal conditions of mutual trust.

15. IANA Considerations

IANA has added the following entries to the ALTO cost map Properties registry, defined in Section 3 of [RFCXXX].

Namespace	Property	Reference
	delay	[RFCxxxx], Section 3
	jitter	[RFCxxxx], Section 4
	pktloss	[RFCxxxx], Section 5
	hopcount	[RFCxxxx], Section 6
	bandwidth	[RFCxxxx], Section 7
	maxbw	[RFCxxxx], Section 8
	maxresbw	[RFCxxxx] Section 9
	unresdbw	[RFCxxxx], Section 10
	residbw	[RFCxxxx], Section 11
	availbw	[RFCxxxx], Section 12
	utilbw	[RFCxxxx], Section 13

16. References

16.1. Normative References

- [ALTO] Alimi, R., "ALTO Protocol", ID draft-ietf-alto-protocol-16, May 2013.
- [BGP-LS] Gredler, H., "North-Bound Distribution of Link-State and TE Information using BGP", ID draft-ietf-idr-ls-distribution-03, May 2013.
- [BGP-PM] Wu, Q., "BGP attribute for North-Bound Distribution of Traffic Engineering (TE) performance Metrics", ID draft-wu-idr-te-pm-bgp-02, October 2013.

- [ISIS-TE] Giacalone, S., "ISIS Traffic Engineering (TE) Metric Extensions", ID draft-ietf-isis-te-metric-extensions-01, October 2013.
- [OSPF-TE] Giacalone, S., "OSPF Traffic Engineering (TE) Metric Extensions", ID draft-ietf-ospf-te-metric-extensions-04, June 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.
- [RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.
- [RFC5234] Crocker, D., "Augmented BNF for Syntax Specifications: ABNF", RFC 5234, January 2008.

16.2. Informative References

- [ALTO-DEPLOYMENT] Stiemerling, M., Kiesel, S., Previdi, S., and M. Scharf, "ALTO Deployment Considerations", ID draft-ietf-alto-deployments-08, October 2013.
- [RFC6390] Clark, A. and B. Claise, "Framework for Performance Metric Development", RFC 6390, July 2011.

Authors' Addresses

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 210012
China

Email: sunseawq@huawei.com

Y. Richard Yang
Yale University
51 Prospect St
New Haven, CT 06520
USA

Email: yry@cs.yale.edu

Young Lee
Huawei
1700 Alma Drive, Suite 500
Plano, TX 75075
USA

Email: leeyoung@huawei.com

Dhruv Dhody
Huawei
Leela Palace
Bangalore, Karnataka 560008
INDIA

Email: dhruv.ietf@gmail.com

Sabine Randriamasy
Alcatel-Lucent
Route de Villejust
Nozay 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: January 2, 2015

G. Bernstein
Grotto Networking
Y. Lee
Huawei
W. Roome
M. Scharf
Alcatel-Lucent
Y. Yang
Yale University
July 1, 2014

ALTO Topology Extensions
draft-yang-alto-topology-03.txt

Abstract

The Application-Layer Traffic Optimization (ALTO) Service has defined Network and Cost maps to provide basic network information. In this document, we discuss an initial design to provide abstracted graph representations of network topology, motivated by a basic application use case of multi-flow scheduling. The design is based on the ALTO WG discussions at IETF 89.

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 2, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Review: the Base Single-Node Representation	4
3. The Multi-flow Scheduling Use Case	4
4. Path-Vector as Cost Metric Representation	5
5. Topology using Opaque Network Elements	8
6. Topology using a Graph (Node-Edge) Representation	9
7. Graph Transformations to Build Topology Representation for Applications	13
8. Operations on Exported Topology	14
9. Security Considerations	14
10. IANA Considerations	14
11. Acknowledgments	14
12. References	14
12.1. Normative References	14
12.2. Informative References	14
Authors' Addresses	15

1. Introduction

Topology is a basic information component that a network can provide to network management tools and applications. Example tools and applications that can utilize network topology include traffic engineering, network services (e.g., VPN) provisioning, PCE, application overlays, among others [RFC5693,I-D.amante-i2rs-topology-use-cases, I-D.lee-alto-app-net-info-exchange].

A basic challenge in exposing network topology is that there can be multiple representations of the topology of the same network infrastructure, and each representation may be better suited for its own set of deployment scenarios. For example, the current ALTO base protocol [I-D.ietf-alto-protocol] is designed for a setting of

exposing network topology using the extreme "my-Internet-view" representation, which abstracts a whole network as a single node that has a set of access ports, with each port connects to a set of endhosts. The base protocol refers to each access port as a PID. This "single-node" abstraction achieves simplicity and provides flexibility. A problem of this abstraction, however, is that the base protocol as currently defined does not provide enough information for use cases such as multi-flow scheduling (see Section 2).

An opposite of the single-node representation is the complete raw topology, spanning across multiple layers, to include all details of network states such as endhosts attachment, physical links, physical switch equipment, and logical structures (e.g., LSPs) already built on top of physical infrastructural devices. A problem of the raw topology representation, however, is that its exposure may violate privacy constraints. Also, a large raw topology may be overwhelming and unnecessary for specific applications. Since the focus of ALTO is general applications which do not want or need to understand detailed raw topology represented in RIB, or details of routing protocols, raw topology does not appear to be a good fit for ALTO.

The objective of this document is to specify a new type of ALTO Information Resources, which provide abstracted graph representations of a network to provide only enough information for applications. We call such Information Resources ALTO Topology Maps, or Topology Maps for short. Different from the base single-node abstraction, a Topology Map includes multiple network nodes. Different from the raw topology representation that uses real network nodes, Topology Maps use abstract nodes, although they should be constructed from the real, raw topology, in order to provide grounded information. The design of this document is based on the ALTO WG discussions at IETF 89, with summary slides at <http://tools.ietf.org/agenda/89/slides/slides-89-alto-2.pdf>.

The organization of this document is organized as follows. We first review the ALTO base protocol in Section 2. Then in Section 3, we give the multi-flow scheduling use case as an example. In Section 4, we specify path vector as a key component to handle multi-flow scheduling. In Sections 5 and 6, we give two graph representations complete the design. Section 7 gives a framework of topology transformations to help with the understanding of deriving multiple representations of the topology of the same network infrastructure, for applications.

2. Review: the Base Single-Node Representation

We distinguish between endhosts and the network infrastructure of a network. Endhosts are sources and destinations of data that the network infrastructure carries. The network itself is neither the source nor the destination of data.

For the given network, it provides "access ports" or access points where digital signal from endhosts enter and leave the network. One should understand "access ports" in a generic sense. For example, an access port can be a physical Ethernet port connecting to a specific endhost, or it can be a port connecting to a CE which connects to a large number of endhosts. Let AP be the set of access ports (AP) that the network provides.

A high-level abstraction of a network topology is only the set AP, and one can visualize the network as a single, abstract node with the set AP of access ports attached. At each ap in AP, a set of endhosts can be reached as destinations. Let $\text{dest}(\text{ap})$ denote the set of endhosts reachable at ap.

There can be multiple ways to partition the set AP. Each partition is called a Network Map. Given a complete partition of AP, the ALTO base protocol introduces PID to represent each partition subset. The ALTO base protocol then conveys the pair-wise connection properties from one PID to another PID through the "single-node". This is the Cost Map.

3. The Multi-flow Scheduling Use Case

There are use cases where simple Cost Metric cannot convey enough information to the applications about pair-wise connection properties from one PID to another PID. See [I-D.bernstein-alto-topo] for a survey of use-cases where extended network topology information is needed. This document uses a simple use case to illustrate the idea.

Consider an application overlay which needs to schedule the traffic among a set of endhost source-destination pairs, say $\text{eh1} \rightarrow \text{eh2}$, and $\text{eh3} \rightarrow \text{eh4}$. Simple Cost Metric such as 'available bw' for $\text{eh1} \rightarrow \text{eh2}$ and $\text{eh3} \rightarrow \text{eh4}$ may not reflect whether the two paths for $\text{eh1} \rightarrow \text{eh2}$ and $\text{eh3} \rightarrow \text{eh4}$ share a bottleneck.

More concretely, assume that the network has 7 switches (sw1 to sw7) forming a dumb-bell topology. Switches sw1/sw3 provide access on one side, sw2/sw4 provide access on the other side, and sw5-sw7 form the backbone. Endhosts eh1 to eh4 are connected to access switches sw1 to sw4 respectively. Assume that the bandwidth of each link is 100

Mbps. Assume that the network is abstracted with 4 PIDs, with each representing the hosts at one access switch.

Now, consider a Cost Map providing end-to-end available bandwidth. There can be two possible interpretations on the semantics of the value of $PID_i \rightarrow PID_j$ reported by the Cost Map: (1) it represents reserved bandwidth from $PID_i \rightarrow PID_j$, or (2) it represents possible bandwidth for $PID_i \rightarrow PID_j$, if no other applications use shared resources. The common understanding is (2), just as when we look at the number of available seats on a flight.

Assume that the application receives from the Cost Map that both $PID_1 \rightarrow PID_2$ and $PID_3 \rightarrow PID_4$ have bandwidth 100 Mbps. It cannot determine that if it schedules the two flows together, whether it will obtain a total of 100 Mbps or 200 Mbps. This depends on whether the flows share a bottleneck:

- o If $PID_1 \rightarrow PID_2$ and $PID_3 \rightarrow PID_4$ use different paths, for example, when the first uses $sw_1 \rightarrow sw_5 \rightarrow sw_7 \rightarrow sw_2$, and the second uses $sw_3 \rightarrow sw_5 \rightarrow sw_6 \rightarrow sw_7 \rightarrow sw_4$. Then the application will obtain 200 Mbps.
- o If $PID_1 \rightarrow PID_2$ and $PID_3 \rightarrow PID_4$ share the bottleneck, for example, when both use the direct link $sw_5 \rightarrow sw_7$, then the application will obtain only 100 Mbps.

To distinguish the two possible cases, the network needs to provide more details.

4. Path-Vector as Cost Metric Representation

A key component to address the problem in the preceding section is to introduce path vectors as a Cost Metric, which is a set of path vectors from a source PID to a destination PID, where each path vector is a sequence (array) of network elements.

A schema for introducing path vectors in Cost Maps is the following extension of Section 11.2.3.6 of [I-D.ietf-alto-protocol]:

```
object {  
  cost-map.DstCosts.JSONValue -> JSONString<0,*>;  
  meta.cost-mode = "path-vector";  
} InfoResourcePVCostMap : InfoResourceCostMap;
```

Specifically, the preceding specifies that `InfoResourcePVCostMap` extends `InfoResourceCostMap`. The body specifies that the first extension is achieved by changing the type of `JSONValue` defined in `DstCosts` of `cost-map` to be an array of `JSONString`; the second extension is that the `cost-mode` of meta MUST be "path-vector".

An example Cost Map using path-vector is the following:

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



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

{
  "meta" : {
    "dependent-vtags" : [
      { "resource-id": "my-default-network-map",
        "tag": "3ee2cb7e8d63d9fab71b9b34cbf764436315542e"
      },
      { "resource-id": "my-topology-map", // See below
        "tag": "4xee2cb7e8d63d9fab71b9b34cbf76443631554de"
      }
    ],
    "cost-type" : { "cost-mode" : "path-vector"
  }
},

  "cost-map" : {
    "PID1": { "PID1":[],
              "PID2":["ne56", "ne67"],
              "PID3":[],
              "PID4":["ne57"]
            },
    "PID2": { "PID1":["ne75"],
              "PID2":[],
              "PID3":["ne75"],
              "PID4":[]
            },
    "PID3": { "PID1":[],
              "PID2":["ne57"],
              "PID3":[],
              "PID4":["ne57"]
            },
    "PID4": { "PID1":["ne75"],
              "PID2":[],
              "PID3":["ne75"],
              "PID4":[]
            }
  }
}
```

The example illustrates that there are two key extensions to the ALTO base protocol:

- o It introduces a new "cost-mode" named path vector.

- o To indicate the resource that provides information on the elements of path vectors (e.g., ["ne5", "ne67"] for the path vector from PID1 to PID2, it introduces a new dependency. In the example, we indicate that it is a resource named "my-topology-map".

5. Topology using Opaque Network Elements

A missing piece to complete the preceding design is how to represent the information resource that provides information on the elements of the path vectors. One simple approach is to introduce simple network element property maps, which provide a mapping from a network element to its properties such as bandwidth or shared risk link group (srlg).

A schema for the Network Element Property Map can be:

```
object-map {  
  JSONString -> NetworkElementProperties; // name to properties  
} NetworkElementMapData;  
  
object-map {  
  JSONString bw;  
  JSONString srlg<0,*>;  
} NetworkElementProperties;
```

An example Network Element Property Map:

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

```

HTTP/1.1 200 OK
Content-Length: TBD
Content-Type: application/alto-nepmap+json

{
  "meta" : {
    "vtag": {
      "resource-id": "my-topology-map",
      "tag": "da65eca2eb7a10ce8b059740b0b2e3f8eb1d4785"
    }
  },
  "nep-map" : {
    "ne57" : { "bw" : 100, "srlg" : [1, 3]}, // link sw5->sw7
    "ne75" : { "bw" : 100, "srlg" : [1, 3]}, // link sw7->sw5
    "ne56" : { "bw" : 100, "srlg" : [1]},    // link sw5->sw6
    "ne65" : { "bw" : 100, "srlg" : [1]},    // link sw6->sw5
    "ne67" : { "bw" : 100, "srlg" : [3]},    // link sw6->sw7
    "ne76" : { "bw" : 100, "srlg" : [3]},    // link sw7->sw6
  }
}

```

An advantage of the representation is that it does not need to distinguish network nodes vs network links.

6. Topology using a Graph (Node-Edge) Representation

A potential problem of the path vector representation is its lacking of compactness. For example, suppose a network has N PIDs, then it will need to represent $N * (N-1)$ paths, if each source-destination pair has one path computed using a shortest-path algorithm. On the other hand, the underlying graph may have only $O(F * N)$ elements, where F is the average degree of the topology.

A graph representation of the example topology in the preceding section has three components:

1. PIDs: {AP1, AP2, AP3}, which is already defined by base ALTO;
2. Nodes: {s1, s2, s3}; and
3. Links: {AP1 -- s1, AP2 -- s2, AP3 -- s3, s1 -- s2, s2 -- s3, s1 -- s3}.

A schema for the graph (node-edge) representation, based on the types already defined in the base ALTO protocol, is the following:

```
object {
  TopologyMapData topology-map;
} InfoResourceTopologyMap : ResponseEntityBase;

object {
  NodeMapData nodes;
  LinkMapData links;
} TopologyMapData;

object-map {
  JSONString -> NodeProperties; // node name to properties
} NodeMapData;

object {
  JSONString type;
  ...
} NodeProperties;

object-map {
  JSONString -> LinkProperties; // link name to properties
} LinkMapData;

object {
  JSONString src;
  JSONString dst;
  JSONString type;
  CostValue costs<0,*>;
} LinkProperties;

object {
  CostMetric metric;
  JSONValue value; // value type depends on metric type
} CostValue;
```

One complexity of the graph representation is multicast/broadcast links. Assume that the link from sw5 -> sw7 is actually a wireless link and the application may benefit in knowing that sw5 -> sw7 and sw5 -> sw6 can be active simultaneously. In other words, sw5 -> [sw6, sw7] is a broadcast link. Knowing such links can be beneficial in settings such as wireless opportunistic routing.

An example using the schema:

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

```
HTTP/1.1 200 OK
Content-Length: TBD
Content-Type: application/alto-topologymap+json
```

```
{
  "meta" : {
    "vtag" : {
      "resource-id" : "my-topology-map",
      "tag" : "da65eca2eb7a10ce8b059740b0b2e3f8eb1d4785"
    }
  },
  "topology-map" : {
    "nodes" : {
      "sw1" : {"type" : "switch"},
      "sw2" : {"type" : "switch"},
      "sw3" : {"type" : "switch"},
      "sw4" : {"type" : "switch"},
      "sw5" : {"type" : "switch"},
      "sw6" : {"type" : "switch"},
      "sw7" : {"type" : "switch"}
    },
    "links" : {
      "e1" : {"src" : PID1,
              "dst" : "sw1",
              "type" : "bidirected";
              "costs" : [
                {"cost-metric" : "availbw", "value" : 100},
                {"cost-metric" : "srlg", value : [1, 3]}
              ]
            },
      "e2" : {"src" : PID2,
              "dst" : "sw2",
              ...
            },
      "e3" : {"src" : PID3,
              "dst" : "sw3",
              ...
            },
      ...
    }
  }
}
```

```

    "e4" : { "src" : PID4,
             "dst" : "sw4",
             ...
    },
    "e15" : { "src" : "sw1",
              "dst" : "sw5",
              ...
    },
    "e35" : { "src" : "sw3",
              "dst" : "sw5",
              ...
    },
    "e27" : { "src" : "sw2",
              "dst" : "sw7",
              ...
    },
    "e47" : { "src" : "sw4",
              "dst" : "sw7",
              ...
    },
    "e57" : { "src" : "sw5",
              "dst" : "sw7",
              ...
    },
    "e56" : { "src" : "sw5",
              "dst" : "sw6",
              ...
    },
    "e67" : { "src" : "sw6",
              "dst" : "sw7",
              ...
    }
  }
}
}

```

Note that the preceding Graph Representation does not provide path informatino. Hence, it should be used with the path-vector representation, or in constraint-light settings where networks use simple, public algorithms to compute routing and hence no need to provide the path vectors explicitly.

7. Graph Transformations to Build Topology Representation for Applications

The preceding sections give a top-down derivation. In this section we give a graph transformation framework to build the schema from a raw topology $G(0)$. The network conducts transformations on $G(0)$ to obtain other topologies, with the following objectives:

1. Simplification: $G(0)$ may have too many details that are unnecessary for the receiving app (assume intradomain); and
2. Preservation of privacy: there are details that the receiving app should not be allowed to see; and
3. Conveying of logical structure (e.g., MPLS paths already computed); and
4. Conveying of capability constraints (the network can have limitations, e.g., it uses only shortest path routing); and
5. Allow modular composition: path from one point to another point is delegated to another app.

The transformation of $G(0)$ is to achieve/encode the preceding. For conceptual clarity, we assume that the network uses a given set of operators. Hence, given a sequence of operations and starting from $G(0)$, the network builds $G(1)$, to $G(2)$, ...

Below is a list of basic operators that the network may use to transform from $G(n-1)$ to $G(n)$:

- o O1: Deletion of a switch/port/link from $G(n-1)$;
- o O2: Switch aggregation: a set V_s of switches are merged as one new (logical) switch, links/ports connected to switches in V_s are now connected to the new logical switch, and then all switches in V_s are deleted;
- o O3: Path representation: For a given extra path from A to R1 to R2 ... to B in $G(n-1)$, a new (logical) link A \rightarrow B is added; if the constraint is that A \rightarrow must use the path, it will be put into the Overlay;
- o O4: Switch split: A switch s in $G(n-1)$ becomes two (logical) switches s_1 and s_2 . The links connected to s_1 is a subset of the original links connected to s ; so is s_2 .

8. Operations on Exported Topology

Going beyond the basic topology exposure from the network and applications/tools, we anticipate that applications and tools can derive results and feed to topology. In particular, we consider the following operations:

- o Instantiation of app guidance in real network: The details of instantiation will be outside the scope of this document. Example protocols include PCEP Extensions for Stateful PCE [I-D.ietf-pce-stateful-pce], RSVP LSP's and their associated characteristics, (i.e.: head and tail-end LSR's, bandwidth, priority, preemption, etc.). The reason that we choose the preceding operator set is that they are "implementable".
- o We also anticipate topology guided mapping of other data: to allow applications to subscribe to statistics and link status from the derived topology.

9. Security Considerations

This document has not conducted its security analysis.

10. IANA Considerations

This document does not specified its IANA considerations, yet.

11. Acknowledgments

The author thanks discussions with Erran Li, Tianyuan Liu, Andreas Voellmy, Haibin Song, and Yan Luo.

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.

12.2. Informative References

- [I-D.amante-i2rs-topology-use-cases]
Amante, S., Medved, J., Previdi, S., and T. Nadeau,
"Topology API Use Cases", draft-amante-i2rs-topology-use-cases-00 (work in progress), February 2013.

[I-D.ietf-alto-protocol]

Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-17 (work in progress), July 2013.

[I-D.lee-alto-app-net-info-exchange]

Lee, Y., Bernstein, G., Choi, T., and D. Dhody, "ALTO Extensions to Support Application and Network Resource Information Exchange for High Bandwidth Applications", draft-lee-alto-app-net-info-exchange-02 (work in progress), July 2013.

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

Authors' Addresses

Greg Bernstein
Grotto Networking
Fremont, CA
USA

Email: gregb@grotto-networking.com

Young Lee
Huawei
TX
USA

Email: leeyoung@huawei.com

Wendy Roome
Alcatel-Lucent Technologies/Bell Labs
600 Mountain Ave, Rm 3B-324
Murray Hill, NJ 07974
USA

Phone: +1-908-582-7974
Email: w.roome@alcatel-lucent.com

Michael Scharf
Alcatel-Lucent Technologies
Germany

Email: michael.scharf@alcatel-lucent.com

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

Email: yry@cs.yale.edu