

ALTO
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2013

S. Kiesel
University of Stuttgart
M. Stiernerling
NEC Europe Ltd.
N. Schwan
Stuttgart, Germany
M. Scharf
Alcatel-Lucent Bell Labs
H. Song
Huawei
October 22, 2012

ALTO Server Discovery
draft-ietf-alto-server-discovery-05

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 specifies a procedure for resource consumer initiated ALTO server discovery, which can be used if the ALTO client is embedded in the resource consumer.

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 April 25, 2013.

Copyright Notice

Copyright (c) 2012 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 Procedure Overview | 5 |
| 3. ALTO Server Discovery Procedure Specification | 6 |
| 3.1. Step 1: Retrieving the Domain Name | 6 |
| 3.1.1. Step 1, Option 1: User input | 6 |
| 3.1.2. Step 1, Option 2: DHCP | 6 |
| 3.2. Step 2: U-NAPTR Resolution | 6 |
| 4. Deployment Considerations | 8 |
| 4.1. Issues with Home Gateways | 8 |
| 4.2. Issues with Multihoming, Mobility and Changing IP Addresses | 8 |
| 5. IANA Considerations | 10 |
| 6. Security Considerations | 11 |
| 6.1. General | 11 |
| 6.2. For U-NAPTR | 11 |
| 7. References | 13 |
| 7.1. Normative References | 13 |
| 7.2. Informative References | 13 |
| Appendix A. Contributors List and Acknowledgments | 15 |
| Authors' Addresses | 16 |

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]. ALTO is realized by a client-server protocol, see requirement AR-1 in [RFC6708]. Before an ALTO client can ask for guidance it needs to discover one or more ALTO servers that can provide suitable guidance.

This document specifies a procedure for resource consumer initiated ALTO server discovery, which 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. A significantly more complex approach, which tries to meet requirement AR-33, i.e., third-party ALTO server discovery, is addressed in [I-D.kist-alto-3pdisc].

The ALTO protocol specification [I-D.ietf-alto-protocol] is based on HTTP and expects the discovery procedure to yield an HTTP(S) URI. Therefore, this procedure is based on U-NAPTR [RFC4848]. It tries to directly find one or more ALTO server(s) that can give suitable guidance to the ALTO client. Other schemes, such as discovering a random ALTO server (which might not be able to give suitable guidance to the client in question) and asking it to redirect the client to a better server, are not considered in this document.

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 Procedure Overview

The ALTO server discovery procedure is performed in two steps:

1. A DNS suffix is yielded, either by manual input or by means of DHCP.
2. This DNS suffix is used for an U-NAPTR lookup yielding the URI. Further DNS lookups may be necessary to determine the ALTO server's IP address(es).

The primary means for retrieving the DNS suffix is DHCP. However, there may be situations where DHCP is not available or does not return a suitable value. Furthermore, there might be situations in which the user wishes to override the value that could be retrieved from DHCP. In these situations, manual input may be used.

Typically, but not necessarily, the DNS suffix is the domain name in which the client is located, i.e., a PTR lookup on the client's IP address would yield a similar name. However, due to the widespread use of network address translation (NAT), trying to determine the DNS suffix through a PTR lookup on the client's IP address is not recommended.

3. ALTO Server Discovery Procedure Specification

A already outlined in Section 2 the ALTO server discovery procedure is performed in two steps, which will be specified in Section 3.1 and Section 3.2, respectively.

3.1. Step 1: Retrieving the Domain Name

3.1.1. Step 1, Option 1: User input

A user may want to use a third party ALTO service instance. Therefore we allow the user to specify a DNS suffix on its own, for example in a config file option. The DNS suffix given by the user is combined with the IP address of the resource consumer to allow the third party ALTO service to direct the client to a suitable ALTO server based on the location of the client. A possible DNS suffix entered by the user may be:

myaltoprovider.org

In case no ALTO NAPTR records are found, we consider the discovery process based on user input as failed. A client MAY try to continue with DHCP (see below). If DHCP-based discovery succeeds the client SHOULD inform the user that the user input has been ignored and replaced by information retrieved from the network.

3.1.2. Step 1, Option 2: DHCP

As a second option network operators MAY configure the domain name to be used for service discovery within an access network using DHCP. RFC 5986 [RFC5986] defines DHCP IPv4 and IPv6 access network domain name options that identify a domain name that is suitable for service discovery within the access network. The ALTO server discovery procedure uses these DHCP options to retrieve the domain name as an input for the U-NAPTR resolution. One example could be:

myisp.com

3.2. Step 2: U-NAPTR Resolution

The ALTO protocol specification [I-D.ietf-alto-protocol] expects that the ALTO discovery procedure yields the HTTP(S) URI of the ALTO server's Information Resource Directory, which gives further information about the capabilities and services provided by that ALTO server. The first step of the ALTO server discovery procedure (see Section 3.1) yielded an U-NAPTR/DDDS (URI-Enabled NAPTR/Dynamic Delegation Discovery Service) [RFC4848] application unique strings, in the form of a DNS name. An example is "example.com".

In the second step, the ALTO Server discovery procedure needs to use the U-NAPTR [RFC4848] specification described below to obtain a URI (indicating host and protocol) for the ALTO server's Information Resource Directory. In this document, only the HTTP and HTTPS URL schemes are defined, as the ALTO protocol specification defines the access over both protocols, but no other [I-D.ietf-alto-protocol]. Note that the HTTP URL can be any valid HTTP(s) URL, including those containing path elements.

The following two DNS entries show the U-NAPTR resolution for "example.com" to the HTTPS URL `https://altoserver.example.com/secure/directory` or the HTTP URL `http://altoserver.example.com/directory`, with the former being preferred.

example.com.

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

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

There is a potential that retrieving the domain name or the U-NAPTR lookup itself does not yield to a result, i.e. no ALTO NAPTR record is found. In this case the discovery procedure failed for this interface. It is RECOMMENDED that clients give up the discovery process and wait a period of time before repeating the procedure. Clients MAY repeat the discovery procedure for a different interface instantaneously.

4. Deployment Considerations

4.1. Issues with Home Gateways

Section 3.1.2 describes the usage of a DHCP option. It enables the network operator of the network, in which the ALTO client is located, to provide a DNS suffix. However, this assumes that this particular DHCP option is correctly passed from the DHCP server to the actual host with the ALTO client, and that the particular host understands this DHCP option. This memo assumes the client to be able to understand the proposed DHCP option, otherwise there is no further use of the DHCP option, but the client has to use the other proposed mechanisms.

There are well-known issues with the handling of DHCP options in home gateways. One issue is that unknown DHCP options are not passed through some home gateways, effectively eliminating the DHCP option.

Another well-known issue is the usage of home gateway specific DNS suffixes which "override" the DNS suffix provided by the network operator. For instance, a host behind a home gateway may receive a DNS suffix ".local" instead of "example.com". This suffix is not usable for the server discovery procedure.

In general, the ALTO server discovery SHOULD be based on the IP address that is used to communicate with other peers, i. e., it should return a server that can provide guidance for that address.

4.2. Issues with Multihoming, Mobility and Changing IP Addresses

If the user decides to enter the DNS suffix manually, only one set of ALTO servers will be discovered, irrespectively of multihoming and mobility. Particularly in mobile scenarios this can lead to undesirable results.

The DHCP-based discovery method can discover different sets of ALTO servers 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 ask the ALTO server(s) for guidance that have been discovered for this specific interface and 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.

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.

There are several challenges with DNS on hosts with multiple interfaces [RFC6418], which can affect the ALTO server discovery. If the DNS resolution is performed on the wrong interface, it can return an ALTO server that could provide sub-optimal or wrong guidance. Finding the best ALTO server for multi-interfaced hosts is outside the scope of this document.

When using Virtual Private Network (VPN) connections there is usually no DHCP. The user has to enter the DNS suffix manually. For good optimization results, a DNS suffix corresponding to the VPN concentrator, not corresponding to the user's current location, has to be entered. Similar considerations apply for Mobile IP.

5. IANA Considerations

IANA is requested to register the following U-NAPTR application service tag:

Application Service Tag: ALTO

Intended usage: Identifies a service that provides a Device with its location information.

Defining Publication: The specification contained within this document.

Contact information: The authors of this document

Author/Change controller: The IESG

6. Security Considerations

6.1. General

There are two different failures for the ALTO server discovery, which can both be caused by malicious attacks or by configuration problems, e. g., in case of DNS configuration errors or multi-homed hosts.

First, the discovery might not be able to discover an ALTO server, even if a suitable ALTO server exists. In that case, ALTO guidance will not be used. The resulting application performance and traffic distribution will correspond to a deployment scenario without ALTO guidance. But given that users cannot rely on the availability of an ALTO server, this results in no significant additional security risk.

Second, the discovery procedure may discover a sub-optimal or wrong ALTO server. Such an ALTO server may either not be able to provide information for a given resource consumer (e. g., behind a NAT), thus rendering the ALTO service useless. Alternatively, it may provide sub-optimal or forged information. In the latter case, attackers could try to use ALTO to affect the traffic distribution or the performance of applications. Users may then observe performance problems, and network operators could detect traffic anomalies. A potential counter-measure is to disable the use of the ALTO service.

Security issues of ALTO in general and potential solutions are also discussed in [I-D.ietf-alto-protocol].

6.2. For U-NAPTR

The address of an ALTO server is usually well-known within an access network; therefore, interception of messages does not introduce any specific concerns.

The primary attack against the methods described in this document is one that would lead to impersonation of an ALTO server since a device does not necessarily have a prior relationship with an ALTO server.

An attacker could attempt to compromise ALTO discovery at any of three stages:

1. providing a falsified domain name to be used as input to U-NAPTR
2. altering the DNS records used in U-NAPTR resolution
3. impersonation of the ALTO server

This document focuses on the U-NAPTR resolution process and hence

this section discusses the security considerations related to the DNS handling. The security aspects of obtaining the domain name that is used for input to the U-NAPTR process is described in respective documents, such as [RFC5986].

The domain name that is used to authenticated the ALTO server is the domain name in the URI that is the result of the U-NAPTR resolution. Therefore, if an attacker was able to modify or spoof any of the DNS records used in the DDDS resolution, this URI could be replaced by an invalid URI. The application of DNS security (DNSSEC) [RFC4033] provides a means to limit attacks that rely on modification of the DNS records used in U-NAPTR resolution. Security considerations specific to U-NAPTR are described in more detail in [RFC4848].

An "https:" URI is authenticated using the method described in Section 3.1 of [RFC2818]. The domain name used for this authentication is the domain name in the URI resulting from U-NAPTR resolution, not the input domain name as in [RFC3958]. Using the domain name in the URI is more compatible with existing HTTP client software, which authenticate servers based on the domain name in the URI.

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.
- [RFC3958] Daigle, L. and A. Newton, "Domain-Based Application Service Location Using SRV RRs and the Dynamic Delegation Discovery Service (DDDS)", RFC 3958, January 2005.
- [RFC4848] Daigle, L., "Domain-Based Application Service Location Using URIs and the Dynamic Delegation Discovery Service (DDDS)", RFC 4848, April 2007.

7.2. Informative References

- [I-D.ietf-alto-deployments]
Stiemerling, M. and S. Kiesel, "ALTO Deployment Considerations", draft-ietf-alto-deployments-03 (work in progress), November 2011.
- [I-D.ietf-alto-protocol]
Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-13 (work in progress), September 2012.
- [I-D.kist-alto-3pdisc]
Kiesel, S. and M. Stiemerling, "3rd Party ALTO Server Discovery (3pdisc)", draft-kist-alto-3pdisc-00 (work in progress), July 2012.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.
- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", RFC 5693, October 2009.
- [RFC5986] Thomson, M. and J. Winterbottom, "Discovering the Local Location Information Server (LIS)", RFC 5986, September 2010.
- [RFC6418] Blanchet, M. and P. Seite, "Multiple Interfaces and Provisioning Domains Problem Statement", RFC 6418,

November 2011.

[RFC6708] Kiesel, S., Previdi, S., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", RFC 6708, September 2012.

Appendix A. Contributors List and Acknowledgments

The initial version of this document was co-authored by Marco Tomsu <marco.tomsu@alcatel-lucent.com>.

Hannes Tschofenig provided the initial input to the U-NAPTR solution part. Hannes and Martin Thomson provided excellent feedback and input to the server discovery.

Olafur Gudmundsson provided an excellent DNS expert review on an earlier version of this document.

The authors would also like to thank the following persons for their contribution to this document or its predecessors: Richard Alimi, David Bryan, Roni Even, Gustavo Garcia, Jay Gu, Xingfeng Jiang, Enrico Marocco, Victor Pascual, Y. Richard Yang, Yu-Shun Wang, Yunfei Zhang, Ning Zong.

Michael Scharf is supported by the German-Lab project (<http://www.german-lab.de>) funded by the German Federal Ministry of Education and Research (BMBF).

Martin Stiernerling is partially supported by the CHANGE project (<http://www.change-project.eu>), a research project supported by the European Commission under its 7th Framework Program (contract no. 257422). 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 CHANGE project or the European Commission.

Authors' Addresses

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

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

Martin Stiernerling
NEC Laboratories Europe
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 4342 113
Email: martin.stiernerling@neclab.eu
URI: <http://ietf.stiernerling.org>

Nico Schwan
Stuttgart, Germany

Email: ietf@nico-schwan.de

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

Email: michael.scharf@alcatel-lucent.com
URI: www.alcatel-lucent.com/bell-labs

Haibin Song
Huawei

Email: melodysong@huawei.com

ALTO
Internet-Draft
Intended status: Experimental
Expires: April 25, 2013

S. Kiesel
University of Stuttgart
M. Stiernerling
NEC Europe Ltd.
October 22, 2012

Third-Party ALTO Server Discovery (3pdisc)
draft-kist-alto-3pdisc-01

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 specifies a procedure for third-party ALTO server discovery, which can be used if the ALTO client is not co-located with the actual resource consumer, but instead embedded in a third party such as a peer-to-peer tracker.

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 April 25, 2013.

Copyright Notice

Copyright (c) 2012 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. Third-Party ALTO Server Discovery Procedure Overview | 5 |
| 3. Third-party ALTO Server Discovery Procedure Specification | 6 |
| 3.1. Step 1: Retrieving the Domain Name | 7 |
| 3.2. Step 2: U-NAPTR Resolution | 7 |
| 4. Deployment Considerations | 8 |
| 4.1. IPv4 PTR lookup | 8 |
| 4.1.1. Private customers or very small businesses | 8 |
| 4.1.2. Medium-size customer networks | 8 |
| 4.1.3. Large Customers | 9 |
| 4.2. IPv6 PTR lookup | 9 |
| 5. Operational Considerations | 10 |
| 6. Security Considerations | 11 |
| 7. IANA Considerations | 12 |
| 8. References | 13 |
| 8.1. Normative References | 13 |
| 8.2. Informative References | 13 |
| Appendix A. Contributors List and Acknowledgments | 14 |
| 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]. ALTO is realized by a client-server protocol, see requirement AR-1 in [RFC6708]. Before an ALTO client can ask for guidance it needs to discover one or more ALTO servers that can provide suitable guidance.

For applications that use a centralized resource directory, such as tracker-based P2P applications, the efficiency of ALTO is significantly improved if the ALTO client is embedded in said resource directory instead of the resource consumer (see Section 4.1 of [I-D.ietf-alto-deployments]). The ALTO client embedded into the resource directory asks for guidance on behalf of the resource consumers. To that end, it needs to discover ALTO servers that can give guidance suitable for these resource consumers, respectively. This is called third-party party ALTO server discovery.

This document specifies a procedure for third-party ALTO server discovery. In other words, this document tries to meet requirement AR-33 in [RFC6708]. To some extent, AR-32, i.e., resource consumer initiated ALTO server discovery, can be seen as a special case of third-party ALTO server discovery. For that matter, an ALTO client embedded in a resource consumer would have to figure out its own "public" IP address (e.g., using STUN [RFC5389]), and then perform the procedures described in this document. However, note that a less flexible yet simpler approach for resource consumer initiated ALTO server discovery is specified in [I-D.ietf-alto-server-discovery].

The ALTO protocol specification [I-D.ietf-alto-protocol] is based on HTTP and expects the discovery procedure to yield an HTTP(S) URI. Therefore, this procedure is based on U-NAPTR [RFC4848]. It tries to directly find one or more ALTO server(s) that can give suitable guidance to the ALTO client. Other schemes, such as discovering an arbitrary ALTO server (which might not be able to give suitable guidance to the client in question) and asking it to redirect the client to a better server, are not considered in this document.

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. Third-Party ALTO Server Discovery Procedure Overview

The third-party ALTO server discovery procedure is performed in two steps:

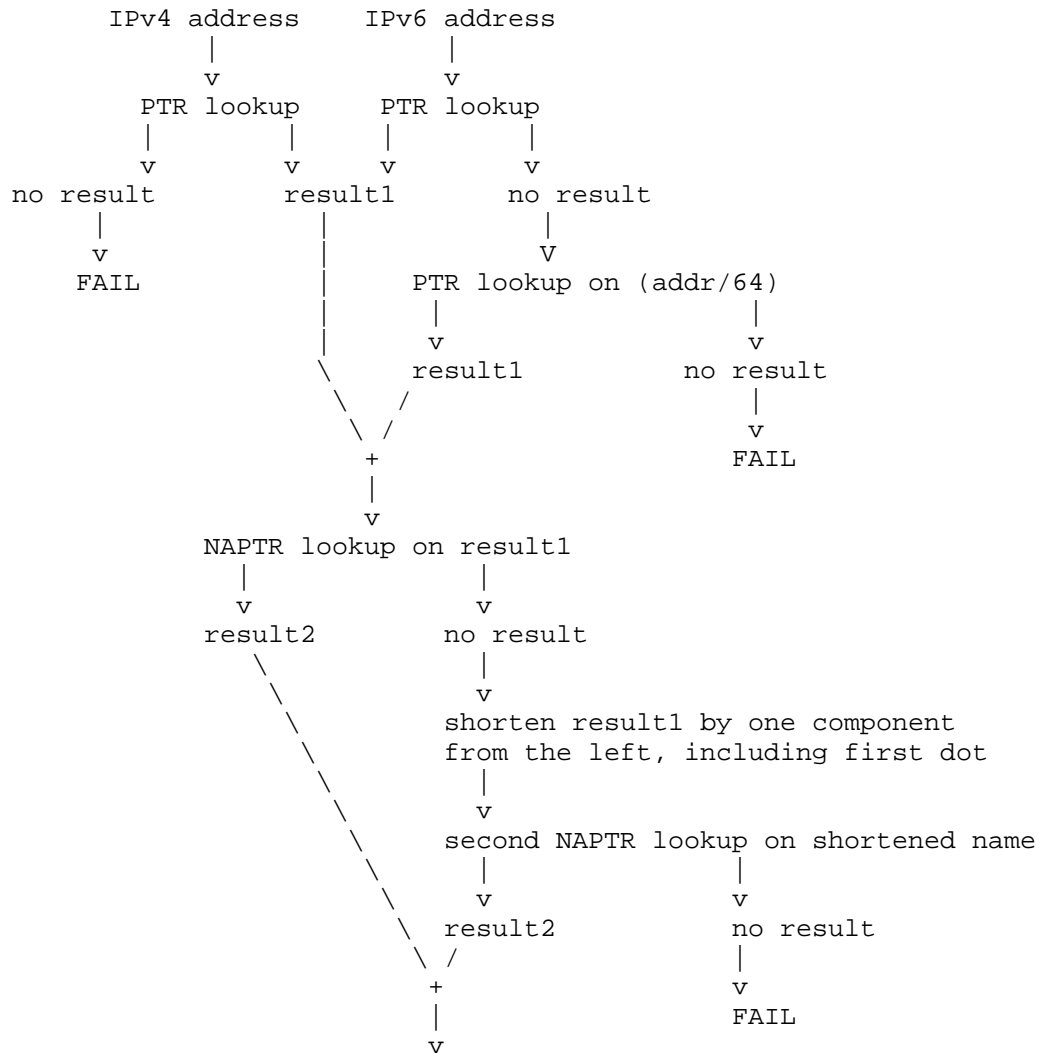
1. A DNS suffix is yielded, by means of a DNS PTR lookup on the resource consumer's IP address (or the "public" IP address of the outermost NAT in front of the resource consumer). This IP address is the source address of application protocol messages arriving at the resource directory.
2. This DNS suffix is used for an U-NAPTR lookup yielding the URI. Further DNS lookups may be necessary to determine the ALTO server's IP address(es).

Typically, but not necessarily, the DNS suffix is the domain name in which the client is located, i.e., a PTR lookup on the client's IP address would yield a similar name. However, due to the widespread use of network address translation (NAT), trying to determine the DNS suffix through a PTR lookup on the client's IP address is not recommended.

Note: Step 1 could be merged with Step 1 of [I-D.ietf-alto-server-discovery] in order to yield a combined draft.

3. Third-party ALTO Server Discovery Procedure Specification

A already outlined in Section 2 the ALTO server discovery procedure is performed in two steps, which will be specified in Section 3.1 and Section 3.2, respectively. The following figure gives an overview:



Third-party ALTO server discovery procedure's output is: result2

3.1. Step 1: Retrieving the Domain Name

Textual specification TBD. See figure above.

Somewhere in the figure, we do a lookup on `addr/64`, which is the "network part" of the IPv6 address computed as follows: `addr/64 := addr & 0xFFFF:FFFF:FFFF:FFFF:0000:0000:0000:0000`.

Note that the algorithm at some point tries to chop one component from the DNS name, but there is no recursion, i.e., no DNS tree walking.

3.2. Step 2: U-NAPTR Resolution

See Step 2 in [I-D.ietf-alto-server-discovery].

4. Deployment Considerations

The mechanism specified in this document needs some configuration effort in order to work properly.

4.1. IPv4 PTR lookup

Especially the domain name retrieved through the reverse DNS lookup (PTR records) and the U-NAPTR entry need to be coordinated. In this section we discuss this configuration for different scenarios.

4.1.1. Private customers or very small businesses

For private customers and very small businesses that are DSL or cable customers often a dynamically assigned IP address is provisioned. Here, the reverse DNS lookup (PTR records) are controlled by the ISP and they point to the ISP's domain, e.g.:

d-c-b-a.dsl.westcoast.my-isp.net

In this case, it would be the responsibility of the respective ISP to provide U-NAPTR entries for the DNS suffix without the endhost part, e.g.:

westcoast.my-isp.net

4.1.2. Medium-size customer networks

The second class of customers have their own DNS domain but only one single upstream ISP, e.g.:

- (1) ISP my-isp.net assigns an IP address a.b.c.d to its customer
- (2) The customer decides that reverse mapping for a.b.c.d should be whatever.customerdomain.com
- (3) If the customer wants to support ALTO, he has to ask the ISP for the URI of the ISP's ALTO server which can give guidance to a.b.c.d. Assume that ISP replies it is http://altoserver.my-isp.net
- (4) The customer establishes a U-NAPTR entry for his domain

```
customerdomain.com.  IN NAPTR 200 10  "u"  "ALTO:http"
                    "!.*!http://altoserver.my-isp.net!"  ""
```

4.1.3. Large Customers

For very large customers with multiple upstream connections we assume that they have their very own traffic optimization policies and thus run their own ALTO server anyway. In this case they need to manage their DNS entries accordingly.

4.2. IPv6 PTR lookup

The IPv6 address space per subnet is much larger than with IPv4 and mechanisms such as privacy extensions allow a host to randomly pick an IP address. Establishing static PTR records for all IPv6 addresses in a /64 prefix is a cumbersome task and unlikely to happen. One alternative is the use of dynamic DNS. Another option is to do without PTR records for individual IP addresses and just have a PTR record for the "network address". The algorithm presented in the figure above can deal with that situation due to the second PTR lookup on (addr/64) if the direct PTR lookup fails.

5. Operational Considerations

TBD.

6. Security Considerations

TBD. See also [I-D.ietf-alto-server-discovery].

7. IANA Considerations

None.

8. References

8.1. Normative References

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

8.2. Informative References

- [I-D.ietf-alto-deployments]
Stiemerling, M. and S. Kiesel, "ALTO Deployment Considerations", draft-ietf-alto-deployments-03 (work in progress), November 2011.
- [I-D.ietf-alto-protocol]
Alimi, R., Penno, R., and Y. Yang, "ALTO Protocol", draft-ietf-alto-protocol-13 (work in progress), September 2012.
- [I-D.ietf-alto-server-discovery]
Kiesel, S., Stiemerling, M., Schwan, N., Scharf, M., and S. Yongchao, "ALTO Server Discovery", draft-ietf-alto-server-discovery-04 (work in progress), July 2012.
- [RFC4848] Daigle, L., "Domain-Based Application Service Location Using URIs and the Dynamic Delegation Discovery Service (DDDS)", RFC 4848, April 2007.
- [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., Stiemerling, M., Woundy, R., and Y. Yang, "Application-Layer Traffic Optimization (ALTO) Requirements", RFC 6708, September 2012.

Appendix A. Contributors List and Acknowledgments

The initial version of this document was co-authored by Marco Tomsu <marco.tomsu@alcatel-lucent.com>.

Hannes Tschofenig provided the initial input to the U-NAPTR solution part. Hannes and Martin Thomson provided excellent feedback and input to the server discovery.

This memo borrows a lot of text from [I-D.ietf-alto-server-discovery], as the 3pdisc was historically part of this memo. Special thanks to Michael Scharf and Nico Schwan.

Martin Stiernerling is partially supported by the CHANGE project (<http://www.change-project.eu>), a research project supported by the European Commission under its 7th Framework Program (contract no. 257422). 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 CHANGE project or the European Commission.

Authors' Addresses

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

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

Martin Stiernerling
NEC Laboratories Europe
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 4342 113
Email: martin.stiernerling@neclab.eu
URI: <http://ietf.stiernerling.org>

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: April 22, 2013

S. Randriamasy, Ed.
Alcatel-Lucent Bell Labs
N. Schwan
October 19, 2012

ALTO Cost Schedule
draft-randriamasy-alto-cost-schedule-02

Abstract

The goal of Application-Layer Traffic Optimization (ALTO) is to bridge the gap between network and applications by provisioning network related information. This allows applications to make informed decisions, for example when selecting a target host from a set of candidates. The ALTO problem statement [RFC5693] considers typical applications as file sharing, real-time communication and live streaming peer-to-peer networks. Recently other use cases focused on Content Distribution Networks and Data Centers have emerged.

The present draft proposes to extend the cost information provided by the ALTO protocol. The purpose is to broaden the decision possibilities of applications to not only decide 'where' to connect to, but also 'when'. This is useful to applications that have a degree of freedom on when to schedule data transfers, such as non-instantaneous data replication between data centers or service provisioning to end systems with irregular connectivity. The draft therefore specifies a new cost mode, called the "schedule" mode. In this mode the ALTO server offers cost maps that contain path ratings that are valid for a given timeframe (e.g. hourly) for a period of time (e.g. a day). Besides the functional time-shift enhancement providing multi-timeframe cost values, the ALTO Cost Schedule also allows to save a number of ALTO transactions and thus resources on the ALTO server and clients. Last, guidance to schedule application traffic can also efficiently help for load balancing and resources efficiency.

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 April 22, 2013.

Copyright Notice

Copyright (c) 2012 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. Use cases for ALTO Cost Schedule | 5 |
| 2.1. Bulk Data Transfer scheduling | 5 |
| 2.2. Endsistemas with limited connectivity or access to datacenters | 6 |
| 2.3. SDN Controller guided access to application endpoints . . . | 8 |
| 3. ALTO Cost Schedule extension | 9 |
| 3.1. Cost Schedule Attributes | 10 |
| 3.1.1. ALTO Cost-Mode: Schedule | 10 |
| 3.2. ALTO Capability: Cost-Scope | 10 |
| 3.2.1. Example of time scope for a cost schedule | 11 |
| 3.3. Example of scheduled information resources in the IRD . . . | 11 |
| 3.3.1. Example scenario and ALTO transaction with a Cost Schedule | 14 |
| 4. IANA Considerations | 15 |
| 4.1. Information for IANA on proposed Cost Types | 16 |
| 4.2. Information for IANA on proposed Endpoint Properties . . . | 16 |
| 5. Acknowledgements | 16 |
| 6. References | 16 |
| 6.1. Normative References | 16 |
| 6.2. Informative References | 17 |
| Authors' Addresses | 17 |

1. Introduction

IETF is currently standardizing the ALTO protocol which aims for providing guidance to overlay applications, that need 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 goal of ALTO is to improve the Quality of Experience (QoE) in the application while simultaneously optimizing resource usage in the underlying network infrastructure.

The ALTO protocol therefore [ID-alto-protocol] specifies a Network Map, which defines groupings of endpoints in a network region (called a PID) as seen by the ALTO server. The Endpoint Cost Service and the Endpoint (EP) Ranking Service then provide rankings for connections between the specified network regions and thus incentives for application clients to connect to ISP preferred endpoints, e.g. to reduce costs imposed to the network provider. Thereby ALTO intentionally avoids the provisioning of realtime information (cmp. ALTO Problem Statement [RFC5693] and ALTO Requirements [RFC5693]), as "Such information is better suited to be transferred through an in-band technique at the transport layer instead". Thus the current Cost Map and Endpoint Cost Service are providing, for a given Cost Type, exactly one rating per link between two PIDs or to an Endpoint. Applications are expected to query one of these two services in order to retrieve the currently valid cost values. They therefore need to plan their ALTO information requests according to the estimated frequency of cost value change. In case these value changes are predictable over a certain period of time and the application does not require immediate data transfer, it would save time to get the whole set of cost values over the period in one ALTO response and using these values to schedule data transfers would allow to optimise the network resources usage and QoE.

In this draft we introduce use cases that describe applications that have a degree of freedom on scheduling data transfers over a period of time, thus they do not need to start a transfer instantaneously on a retrieved request. For this kind of applications we propose to extend the Cost Map and Endpoint Cost Services by adding a schedule on the cost values, allowing applications to time-shift data transfers.

In addition to this functional ALTO enhancement, we expect to further gain by gathering multiple Cost Values for one cost type as one Cost Map reporting on N Cost Values is less bulky than N Cost Maps containing one Cost value each, in addition to reducing N ALTO transactions to a single one. This is valuable for both the storage

of these maps and their transfer. Similar gains can be obtained for the ALTO Endpoint Cost Service.

The remainder of this draft first provides use cases that motivate the need for a 'schedule' cost mode. It then specifies the needed extensions to the ALTO protocol and details some example messages.

2. Use cases for ALTO Cost Schedule

This section introduces use cases showing the benefits of providing ALTO Cost values in 'schedule' mode. Most likely, the ALTO Cost Schedule would be used for the Endpoint Cost Service where a limited set of feasible non real time application Endpoints is already identified, they need to be accessed neither simultaneously nor immediately and their access can be scheduled within a given time period. The Filtered Cost Map service is also applicable as long as the size of the Map is manageable. An ALTO Cost schedule can be used in several ways:

- o the ALTO Server may provide values on past time periods that can be interpreted as historical experience and used to anticipate future cost values in order to schedule transfers of application data or services,
- o the ALTO Server may provide values on present or future time periods that can be interpreted as predictions on cost values and used to schedule transfers of application data or services,
- o the ALTO Server may provide values on time periods covering the past, present and future and logically be all interpreted as predictions and used to schedule transfers of application data or services.

2.1. Bulk Data Transfer scheduling

Some CDNs are prepopulating caches with content before it actually gets available for the user and thus there is a degree of freedom on when the content is transmitted from the origin server to the caching node. Other applications like Facebook or YouTube rely on data replication across multiple sites for several reasons, such as offloading the core network or increasing user experience through short latency. Typically the usage pattern of these data centers or caches follows a location dependent diurnal pattern.

In the examples above data needs to be replicated across the various locations of a CDN provider, leading to bulk data transfers between datacenters. Scheduling these data transfers is a non-trivial task

as the transfer should not infer with the user peak demand to avoid degradation of user experience and to decrease billing costs for the datacenter operator by leveraging off-peak hours for the transfer. This peak demand typically follows a diurnal pattern according to the geographic region of the datacenter. One precondition to schedule transfers however is to have a good knowledge about the demand and link utilization patterns between the different datacenters and networks.

While this usage data today already is gathered and also used for the scheduling of data transfer, provisioning this data gets increasingly complex with the number of CDN nodes and in particular the number of datacenter operators that are involved. For example, privacy concerns prevent that this kind of data is shared across administrative domains. The ALTO Cost Schedule specified later in this document avoids this problem by presenting an abstracted view of time sensitive utilization maps through a dedicated ALTO service to allow CDN operators a mutual scheduling of such data transfers across administrative domains.

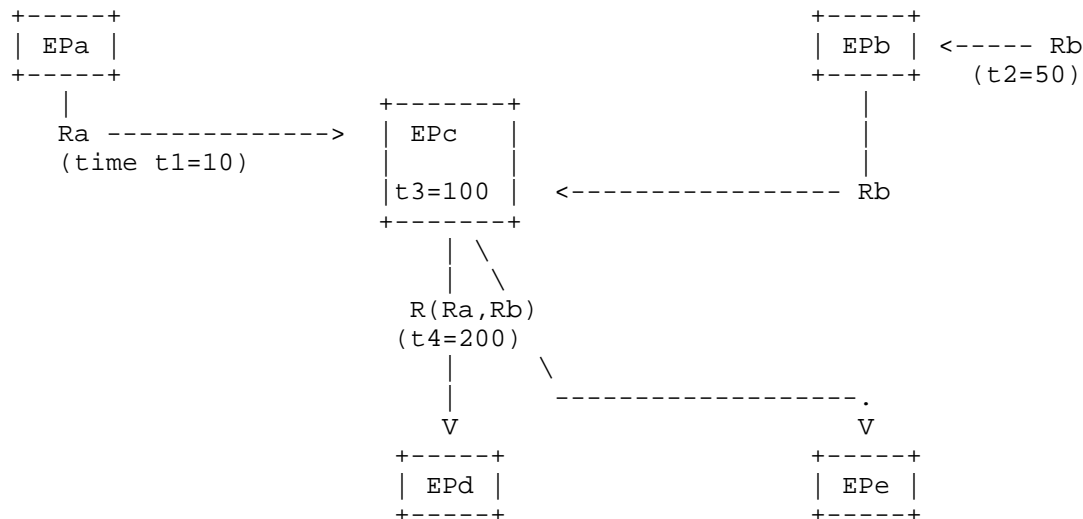
2.2. Endsistemas with limited connectivity or access to datacenters

Another use case that benefits from the availability of multi-timeframe cost information is based on applications that are limited by their connectivity either in time or resources or both. For example applications running on devices in remote locations or in developing countries that need to synchronize their state with a data center periodically, in particular if sometimes there is no connection at all. Example applications is enterprise database update, remote learning, remote computation distributed on several data center endpoints.

Wireless connectivity has a variable quality or may even be intermittent. On the other hand, the connectivity conditions are often predicable. For non real time applications, it is thus desirable to provide ALTO clients with routing costs to connection nodes (i.e. Application Endpoints) over different time periods. This would allow end systems using ALTO aware application clients to schedule their connections to application endpoints.

Another challenge arises with end systems using resources located in datacenters and trading content and resources scattered around the world. For non-real time applications, the interaction with Endpoints can be scheduled at the time slots corresponding to the best possible QoE. For instance, resource Ra downloaded from Endpoint EPa at time t1, Resource Rb uploaded to EPb at time t2, some batch computation involving Ra and Rb done on EPc at time t3 and results R(A,B) downloaded to EPd and EPe at time t4. Example

applications are similar to the ones cited in the previous paragraph.



These examples describe situations where a client has the choice of trading content or resources with several Endpoints and needs to decide with which Endpoint it will trade and at what time. For instance, one may assume that the Endpoints are spread over different time-zones, or have intermittent access. The ALTO Schedule mode specified below allows these clients to retrieve Endpoint cost maps valid for a certain timeframe (e.g. 24 hours), and get a set of values, each applicable on a (e.g. hourly) slot. Thus the application can optimize the needed data transfer according to this information.

Last the ALTO Cost schedule is beneficial to optimizing ALTO transactions themselves. Indeed, let us assume that an Application Client is located in an end system with limited resources and/or has an access to the network that is either intermittent or provides an acceptable QoE in limited but predictable time periods. In that case, it needs to both schedule its resources demanding networking activities and its ALTO requests. Instead of having to figure out when the cost values may change and having to carefully schedule multiple ALTO requests, it could avoid this by relying on Cost Schedule attributes that indicate the time granularity, the validity and time scope of the cost information, together with the time related cost values themselves.

Suppose that for some Cost Types, the ALTO cost values are available

in the "schedule" mode. If the values of Cost type 'routingcost' and/or another time-sensitive Cost Type named for example 'pathoccupationcost' are available in the "schedule" mode for the 24 following the last update, the ALTO Client embedded in the Application Client may query ALTO information on 'routingcost' or 'pathoccupationcost' for these 24 hours, and get a set of values, each applicable to an hour slot. If appropriate Cost Attributes are provided together with the cost values, the Application client also knows the date of their last update. An example ALTO transaction is provided later in this draft.

2.3. SDN Controller guided access to application endpoints

The Software Defined Networking (SDN), see [sdnrg], is a model that attempts to manage and reconfigure networks in a more flexible way in order to better cope with the traffic challenges posed by nowadays resources greedy applications. To this end, one option is "moving the control plane out of the network elements into "controllers", see [SDN charter, <http://www.1-4-5.net/~dmm/sdnrg/sdnrg.html>], that implements the network control and management. The SDN Controllers are deemed to gather the network state information and provide it in an abstracted form to SDN aware applications while gathering their requirements in QoE and exchanging other application "management" information and commands.

The relevance of ALTO to perform a number of SDN functions has been recently highlighted. An ALTO Server can assist an SDN Controller by hosting abstracted network information that can be provided to SDN aware applications via an ALTO Client. It can also assist other SDN Control operations using information in and outside the ALTO scope.

In particular, [article-gslh-alto-sdn] identifies SDN Controller functions that ALTO is well suited to perform: the primitives of Abstraction, Get network topology, Get network resources and Event notification. Additionally, the interaction between ALTO and SDN has been investigated in [draft-xie-alto-sdn] to provide applications with a path selection meeting QoS requirements on bandwidth and delay.

Currently, the base ALTO protocol allows to perform the following SDN services, see [article-gslh-alto-sdn]:

1. Abstraction: through aggregation into PIDs, ranking and a generic cost type.
2. Get network topology: through the Map and the Cost Map Services

3. Get device capabilities: through the Endpoint Property Service.

Another SDN primitive "Get network resources" provides applications with informations allowing them to evaluate the expected QoE. QoE related information includes delay and bandwidth at the application endpoints as well as on the network paths. Such information may be provided via the ALTO Service by proposed extensions of the ALTO protocol that define new ALTO Cost Types allowing to abstract and report QoE to applications.

One key objective of an SDN controller is the ability to balance the application traffic whenever possible. For non real time applications, data and resources transfer can be time shifted, resources availability may often be predicable and last, strong incentives for applications to time shift their traffic may be given by network operators appropriately setting routing cost values at different time values, according to their policy to cope with network occupation over time.

To achieve this objective, the SDN controller can:

1. get the network state history from its controlled network elements through its southbound API
2. possibly derive an estimation or a prediction of these values over given time frames
3. store their abstraction in an ALTO Server in the form of ALTO Cost Schedule values defined for different time periods
4. deliver these values to the SDN applications via the ALTO Endpoint Cost Service, either as history or prediction or as estimations covering both the past and the future.

This way:

- o One one hand, the applications get the best possible QoE, as they can pick the best time for them to access one or more Endpoints,
- o One the other hand the SDN controller achieves load balancing as it may guide the application traffic so as to better distribute the traffic over time, and thus optimize its resources usage.

3. ALTO Cost Schedule extension

One example of non-realtime information that can be provisioned in a 'schedule' is the expected path bandwidth. While the transmission

rate can be measured in real time by end systems, the operator of a data center is in the position of formulating preferences for given paths, at given time periods of given time scales, for example to avoid hotspots due to diurnal usage patterns. The entity managing the ALTO Server values can decide to integrate path bandwidth in the ALTO 'routingcost' metric. However to better highlight the purpose of the cost schedule the remainder of this document will use a Cost Type named 'pathoccupationcost' and assumed to report an abstracted form of available bandwidth. A definition and usage of such a Cost-Type is proposed in [draft-randriamasy-multi-cost-alto].

The usage of a time related cost is rather proactive in that it can be used like a "time table" to figure out the best time to schedule data transfer and also anticipate predictable events including predictable flash crowds. An ALTO Cost Schedule should be viewed as a synthetic abstraction of real measurements that can be historic or be a prediction for upcoming time periods.

3.1. Cost Schedule Attributes

Specifications on the cost "schedule" are proposed here and will be completed in further versions of this draft.

3.1.1. ALTO Cost-Mode: Schedule

The "schedule" mode applies to Costs that are eligible for a single-valued Cost Mode and can also be expressed as such. In that sense, when the "numerical" mode is available for a Cost-Type, the cost expressed in the "schedule" mode is an extension of its expression from one value in the "numerical" mode to an array of several values varying over time.

Types of Cost values such as JSONBool can also be expressed in the "schedule" mode, as states may be "true" or "false" depending on given time periods. It may be expressed as a single value which is either "true" or "false" following a decision rule outside the ALTO protocol.

3.2. ALTO Capability: Cost-Scope

To ensure that the application client uses the NP provided information in the cost schedule in an unambiguous way we define the Cost Scope capability, which defines the validity of the "scheduled" cost values.

For Cost Types whose values are provided in a mode different than 'schedule', the Cost Scope capability is specified by the string "permanent". The Cost Scope attributes provided for the 'schedule'

mode are listed below. The reference time zone for the provided values is UTC.

- o Unit: expresses the time interval applicable to each value. A two element array where the first element is the time unit, ranging from "second" to "year", and the second one the number of units of this duration. For example: '['minute', 5] means that each value is provided on a time interval lasting 5 minutes.
- o Size: the number of values of the cost schedule array,
- o Begin: the index of the first unit in the array,
- o Reference time zone: set to "UTC",
- o Next update: the date at which the sample will be re-computed,
- o Last update: the last re-computation date.

The reference time zone is UTC.

Attributes 'Last update 'and 'Next update' report on the update frequency and age of the information.

3.2.1. Example of time scope for a cost schedule

Let us assume that the metric 'pathoccupationcost' (POC for short) is computed for 24 hours, on time intervals lasting 2 hours, with the first interval starting at 0h00. The ALTO Server thus provides an array 12 values. This information is then used to enable applications to see which time intervals in a day are the most favorable to operate, and which "busy " time intervals should be avoided. If the "Begin" date is past, the application can also use the information to compute statistics or infer a some customized prediction.

3.3. Example of scheduled information resources in the IRD

The example IRD given in this Section includes 2 particular URIs:

- o "http://alto.example.com/endpointcost/lookup", in which the ALTO Server offers several Endpoint Cost Types, including a Cost called "pathoccupationcost" for which the "schedule" Cost Mode is available. The Endpoint Costs available are the "hopcount", "routingcost" and "pathoccupationcost" Cost Types, with the two first ones in the "numerical" Cost Mode and "pathoccupationcost" in the "schedule" Cost Mode.

- o "http://custom.alto.example.com/endpointcost/schedule/lookup", in which the ALTO Server provides the 'routingcost' in both "numerical" and "schedule" modes. This resource is accessible via a separate subdomain called "custom.alto.example.com". The ALTO Client may either get the last update of the 'routingcost' value or request for a previsual sample of 24 values established each for 1 hour. An ALTO Client can discover the services available at "custom.alto.example.com" by successfully performing an OPTIONS request to "http://custom.alto.example.com/endpointcost".

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

```
{
  ... usual ALTO resources ...

  "resources" : [
    {
      "uri" : "http://alto.example.com/endpointcost/lookup",
      "media-types" : [ "application/alto-endpointcost+json" ],
      "accepts" : [ "application/alto-endpointcostparams+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "cost-modes" : [ "numerical", "numerical", "schedule" ],
        "cost-types" : [ "routingcost", "hopcount", "pathoccupationcost" ],
        "cost-scope": [ "permanent", "permanent",
          { "unit": [ "hour", 1 ], "size": 24, "begin": 0,
            "time zone": "UTC",
            "lastupdate": mm/hh/dd/mm/yyyy,
            "nextupdate": mm/hh/dd/mm/yyyy }
        ]
      },
    },
    {
      "uri" : "http://custom.alto.example.com/endpointcost/schedule/lookup",
      "media-types" : [ "application/alto-endpointcost+json" ],
      "accepts" : [ "application/alto-endpointcostparams+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "cost-modes" : [ "numerical", "schedule" ],
        "cost-types" : [ "routingcost", "routingcost" ],
        "cost-scope": [ "permanent",
          { "unit": [ "hour", 1 ], "size": 24, "begin": 0,
            "time zone": "UTC",
            "lastupdate": mm/hh/dd/mm/yyyy,
            "nextupdate": mm/hh/dd/mm/yyyy }
        ]
      },
    }
  ]
}
```

3.3.1. Example scenario and ALTO transaction with a Cost Schedule

Let us assume an Application Client located in an end system with limited resources and having an access to the network that is either intermittent or provides an acceptable quality in limited but possibly predictable time periods. Therefore, it needs to both schedule its resources demanding networking activities and minimize its ALTO transactions.

The Application Client has the choice to trade content or resources with a set of Endpoints of moderate 'routingcost', and needs to decide with which Endpoint it will trade at what time. For instance, one may assume that the Endpoints are spread on different time-zones, or have intermittent access. In this example, the 'routingcost' is assumed constant for the scheduling period and the time sensitive decision metric is the path bandwidth reflected by a Cost type called 'pathoccupationcost'.

The ALTO Client embedded in the Application Client queries ALTO information on 'pathoccupationcost' for the 24 hours following (implicitly) the date of "lastupdate", as this resource is listed in the IRD.

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

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

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

```
{
  "meta" : {},
  "data" : {
    "cost-type" : ["pathoccupationcost"],
    "cost-mode" : ["schedule"],
    "map" : {
      "ipv4:192.0.2.2": {
        "ipv4:192.0.2.89" : [7, ... 24 values],
        "ipv4:198.51.100.34" : [4, ... 24 values],
        "ipv4:203.0.113.45" : [2, ... 24 values]
      }
    }
  }
}
```

4. 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],

4.1. 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.

4.2. 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.

5. Acknowledgements

Thank you to the ALTO WG for fruitful discussions.

Sabine Randriamasy is partially supported by the MEVICO project (<http://www.celtic-initiative.org/Projects/Celtic-projects/Call7/MEVICO/mevico-default.asp>), a research project supported by the European Commission under its 7th Framework Program CELTIC initiative (project no. CP 07-011). 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 MEVICO project or the European Commission.

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

6.2. Informative References

- [ID-alto-protocol]
R. Alimi, R. Penno, Y. Yang, Eds., "ALTO Protocol, draft-ietf-alto-protocol-13.txt", September 2012.
- [article-gslh-alto-sdn]
V. Gurbani, M. Scharf, T. Lakshman, and V. Hilt, "Abstracting network state in Software Defined Networks (SDN) for rendezvous services, IEEE International Conference on Communications (ICC) Workshop on Software Defined Networks (SDN)", June 2012.
- [draft-jenkins-alto-cdn-use-cases-01]
B. Niven-Jenkins (Ed.), G. Watson, N. Bitar, J. Medved, S. Previdi, "Use Cases for ALTO within CDNs, draft-jenkins-alto-cdn-use-cases-01", June 2011.
- [draft-randriamasy-multi-cost-alto]
S. Randriamasy, Ed., B. Roome, N. Schwan, "Multi-Cost ALTO, draft-randriamasy-alto-multi-cost-07", October 2012.
- [draft-xie-alto-sdn]
H. Xie, T. Tsou, D. Lopez, H. Yin, "Use Cases for ALTO with Software Defined Networks, draft-xie-alto-sdn-extension-use-cases-00", June 2012.
- [sdnrg] "Software Defined Network Research Group, <http://trac.tools.ietf.org/group/irtf/trac/wiki/sdnrg>".

Authors' Addresses

Sabine Randriamasy (editor)
Alcatel-Lucent Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

Nico Schwan

Email: ietf@nico-schwan.de

Network Working Group
Internet-Draft
Intended status: Experimental
Expires: April 22, 2013

S. Randriamasy, Ed.
W. Roome
Alcatel-Lucent Bell Labs
N. Schwan
October 19, 2012

Multi-Cost ALTO
draft-randriamasy-alto-multi-cost-07

Abstract

IETF is designing a new service called ALTO (Application Layer traffic Optimization) that includes a "Network Map Service", an "Endpoint Cost Service" and an "Endpoint (EP) Ranking Service" and thus incentives for application clients to connect to ISP preferred Endpoints. These services provide a view of the Network Provider (NP) topology to overlay clients.

The present draft proposes a light way to extend the information provided by the current ALTO protocol 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 lighter than N single cost transactions. All this also helps producing a faster and more robust choice when multiple Endpoints need to be selected.

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 April 22, 2013.

Copyright Notice

Copyright (c) 2012 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 | 5 |
| 2. Application scope and terminology | 6 |
| 3. Uses cases for using multiple costs | 7 |
| 3.1. Use cases for using additional costs | 7 |
| 3.1.1. Delay Sensitive Overlay Applications | 8 |
| 3.1.2. Selection of physical servers involved in virtualized applications | 8 |
| 3.1.3. CDN Surrogate Selection | 9 |
| 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 | 11 |
| 3.2.2. Optimized Filtered Cost Map Service | 11 |
| 3.2.3. Cases of unpredictable Endpoint cost value changes | 12 |
| 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 | 16 |
| 4.2.1. Cost value encoded in JSONArray | 16 |
| 4.2.2. Format update on CostMode and CostType | 16 |
| 4.3. Rules required on object member description | 17 |
| 4.3.1. Rule on cost value order in ALTO reponses | 17 |
| 4.3.2. Rule on mapping for cost-type and cost-mode array members | 17 |
| 4.4. Updates recommended in the object structure | 17 |
| 4.5. Rule recommended on the cost value mode | 17 |
| 4.6. Extended constraints on multi-cost values | 18 |
| 4.6.1. Use cases for mutli-cost multi-operator constraints | 18 |
| 4.6.2. Extended constraints in Multi-Cost ALTO | 19 |
| 5. Protocol extensions for multi-cost ALTO transactions | 19 |
| 5.1. Information Resources Directory | 20 |
| 5.1.1. Example of Multi-Cost specific resources in the IRD | 20 |
| 5.2. Multi-Cost Map Service | 22 |
| 5.2.1. Media Type | 22 |
| 5.2.2. HTTP Method | 22 |
| 5.2.3. Input Parameters | 22 |
| 5.2.4. Capabilities | 22 |
| 5.2.5. Response | 23 |
| 5.2.6. Example | 25 |
| 5.3. Filtered Multi-Cost Map | 25 |
| 5.3.1. Media Type | 26 |
| 5.3.2. HTTP Method | 26 |
| 5.3.3. Input Parameters | 26 |

| | | |
|--------|--|----|
| 5.3.4. | Capabilities | 28 |
| 5.3.5. | Response | 28 |
| 5.3.6. | Example 1 | 29 |
| 5.3.7. | Example 2 | 29 |
| 5.4. | Endpoint Multi-Cost Service | 30 |
| 5.4.1. | Media Type | 31 |
| 5.4.2. | HTTP Method | 31 |
| 5.4.3. | Input Parameters | 31 |
| 5.4.4. | Capabilities | 32 |
| 5.4.5. | Response | 32 |
| 5.4.6. | Example | 33 |
| 6. | IANA Considerations | 34 |
| 6.1. | Information for IANA on proposed Cost Types | 35 |
| 6.2. | Information for IANA on proposed Endpoint Properties | 35 |
| 7. | Acknowledgements | 35 |
| 8. | References | 35 |
| 8.1. | Normative References | 35 |
| 8.2. | Informative References | 36 |
| | Authors' Addresses | 36 |

1. Introduction

IETF is designing 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 current ALTO protocol draft [ID-alto-protocol-11] 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.

There are three parts in this draft. The first part exposes use cases motivating the introduction of new Cost Types and why multi-

cost transactions are useful. The second part specifies the core ALTO protocol extensions that are required or recommended to support requests and responses on multiple Cost Types in one single transaction. These extensions also integrate the discussions within the ALTO Working Group. The third part lists the Multi-Cost ALTO services that can be supported by these extensions.

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.

The 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-11] 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 current specification registers 'routingcost' as a generic measure for routing traffic from a source to a destination. In a similar way the ALTO Endpoint Property Registry ensures uniqueness of ALTO Endpoint Property identifiers and provides references to particular semantics of the allocated 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 "EPCapacity", 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 [ID-ALTO-Requirements13], 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

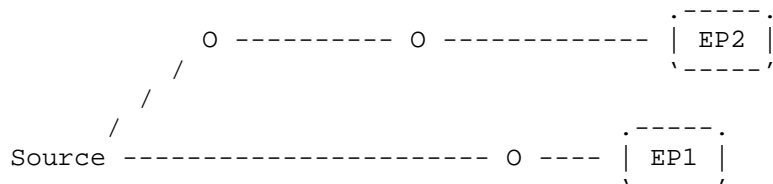
In Figure 1, initially at time T=1, the application has chosen EP1 rather than EP2, despite the higher routing cost, because EP1 has a "better" (lower) 'hopcount' value and despite the higher routing cost and possibly because the application has set a higher weight to 'hopcount'.

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

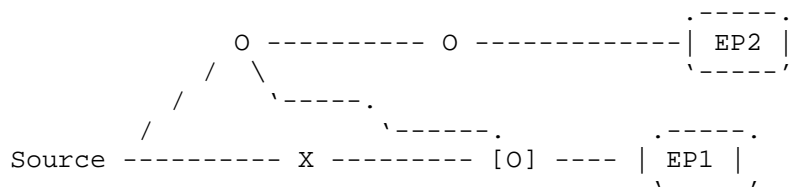
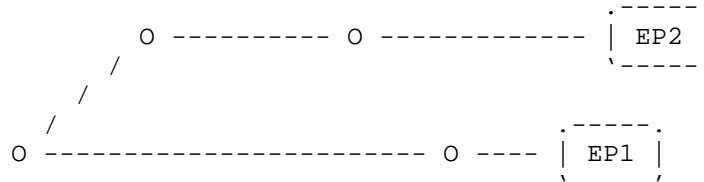


Figure 1: Endpoint re-selection using Multi-Cost ALTO request on updated cost values, upon a change in the route.

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.

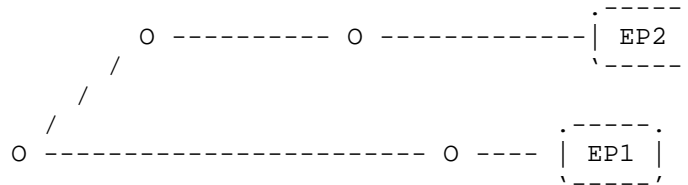


Figure 2: Endpoint selection using 2 Cost Types with joint request on updated cost values and for delay sensitive applications.

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 into 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 the previous version of the ALTO protocol draft, see [ID-alto-protocol]. It partially integrates an update of the current version issued recently, see [ID-alto-protocol-11], that proposes a generic encoding of cost values in the 'JSONValue' data type. The proposed Multi-Cost specifications will be updated according to the outcome of WG discussions.

This section lists and details the proposed changes according to the previous ALTO protocol draft, [ID-alto-protocol] .

If members 'cost-type' and 'cost-mode' of objects InfoResourceCostMap, InfoResourceEndpointCostMap, ReqFilteredCostMap, ReqEndpointCostMap remain specified as single values in the base ALTO protocol, then Multi-Cost specific media types need to be used similarly to those specified in the previous version of this draft, see [draft-randriamasy-alto-multi-cost-05].

4.1. List of ALTO protocol updates required and recommended

The following updates to the current ALTO protocol, see [ID-alto-protocol], are required or recommended to support multi-cost ALTO transactions. The new resulting JSON formats are specified in the next sections.

- o Updates required in the format of objects member(s):
 - * Objects DstCosts (to destination PIDs) and EndpointDstCosts (to destination Endpoints): JSON type of cost value member evolves from JSONNumber to JSONArray.
 - * Objects InfoResourceCostMap, ReqFilteredCostMap, ReqEndpointCostMap, InfoResourceEndpointCostMap: members 'cost-mode' and 'cost-type' have now an array of values rather than a single value.
- o Updates recommended in the object structure:
 - * Objects CostMapCapability and FilteredCostMapCapability: new member giving the maximum number of Cost Types in a response.
- 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 draft, up to version 10

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 2 PIDs or to an Endpoint 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):

```
["routingcost", "hopcount", "quarterlyvaluexxx", "statustring"]
```

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. Format update on CostMode and CostType

In the base protocol, Objects InfoResourceCostMap, ReqFilteredCostMap, ReqEndpointCostMap, InfoResourceEndpointCostMap have members 'cost-mode' and 'cost-type' that list which Cost Type is reported and in which mode this Cost Type is represented.

In Multi-Cost ALTO several Cost Types are used per destination PID or Endpoint, so the member 'cost-type' of these objects must now be an array of values rather than a single value. Likewise, the member 'cost-mode' must now be an array, where each value reports the representation mode of the corresponding index in the 'cost-type' list.

The change on members 'cost-mode' and 'cost-type' from a single value to an array of values are specified in later sections.

4.3. Rules required on object member description

When several cost values are provided, it is necessary to unambiguously specify to which Cost Type each value corresponds and in which mode each value is provided.

4.3.1. 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.

4.3.2. Rule on mapping for cost-type and cost-mode array members

The cost-mode array **MUST** be of the same size as the cost-type array. Each value of this array maps to the Cost Type ID at the same place in the Cost Type array and this value specifies the mode in which the value for this Cost Type is provided.

4.4. Updates recommended in the object structure

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

4.5. Rule recommended on the cost value mode

In multi-cost transactions: when the mode 'numerical' is available for a Cost Type, it **MUST** be the one used to represent the cost values. In any case, the Cost Mode used for each Cost Type **MUST** be exactly specified.

The following example illustrates how this rule can be applied:

Assume the Cost Types array:

```
["routingcost", "hopcount", "quarterlyvaluexxx", "statuststring"]
```

The corresponding Cost Mode array should be (assuming that these modes are supported):

```
["numerical", "numerical", "dynamic", "string"]
```

An example of values is:

```
[23, 6, [21, 9, 4, 12], "medium"]
```

In this example, it is assumed that when the value of a Cost Type is expressed by an array of numbers such as [21, 9, 4, 12], the values in this array are expressed in the 'numerical' mode.

4.6. 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.

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 6.8.2.2.3 "Input parameters" 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)

However, an application

4.6.1. Use cases for mutli-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].

4.6.2. 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.

5. 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.

5.1. 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.

5.1.1. 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 cosy 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

{
  "resources" : [
    {
      .....
      Usual ALTO "single-cost" Services as described in current ALTO Protocol
      .....
    }, {
      "uri" : "http://alto.example.com/multi/costmap",
      "media-types" : ["application/alto-multicostmap+json"],
      "capabilities" : {
        "cost-types" : [ "routingcost", "hopcount" ],
        "cost-modes" : [ "numerical", "numerical" ]
      }
    }, {
      "uri" : "http://alto.example.com/multi/costmap/filtered",
      "media-types" : ["application/alto-multicostmap+json" ],
      "accepts" : ["application/alto-multicostmapfilter+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "max-cost-types" : 3,
        "cost-types" : [ "routingcost", "hopcount", "pathoccupationcost" ],
        "cost-modes" : [ "numerical", "numerical", "numerical" ]
      }
    }, {
      "uri" : "http://alto.example.com/multi/endpointmulticost/lookup",
      "media-types" : [ "application/alto-endpointmulticost+json" ],
      "accepts" : [ "application/alto-endpointmulticostparams+json" ],
      "capabilities" : {
        "cost-constraints" : true,
        "max-cost-types" : 2,
        "cost-types" : [ "routingcost", "hopcount", "status" ],
        "cost-modes" : [ "numerical", "numerical", "string" ]
      }
    }
  ]
}
```


5.2. 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, the array is a list of nulls -- then the ALTO server MAY omit the entry for that source/destination pair.

5.2.1. Media Type

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

5.2.2. HTTP Method

This resource is requested using the HTTP GET method.

5.2.3. Input Parameters

None.

5.2.4. Capabilities

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

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

with members

cost-types The Cost Types (Section 5.XX) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

cost-modes The Cost Modes (Section 5.XX) supported for each of the supported Cost Types listed in the array 'cost-types'. This array MUST have the same size as the array 'cost-types', and each member of this array MUST give the mode of the Cost Type at that index.

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

An ALTO Server supporting the Multi-Cost Map service, MUST support the Cost mode 'numerical' for all supported Cost Types encoded with the 'JSONNumber' type.

5.2.5. Response

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

```

    object DstMultiCosts {
      JSONArray [PIDName];
      ...
    };

    object {
      DstMultiCosts [PIDName]<0..*>;
      ...
    } MultiCostMapData;

    object {
      CostType      cost-type<0..*>;
      CostMode      cost-mode<0..*>;
      JSONString    map-vtag;
      MultiCostMapData map;
    } InfoResourceMultiCostMap;

```

with members:

cost-mode Array of Cost Modes (Section xxx) used in the Cost Map. This array MUST have the same size as the array 'cost-types'. where each member of the cost-mode array is the Cost Mode used for the Cost Type at the same place in the array.

cost-type The array of Cost Types (Section xxx) used in the Cost Map.

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

map The Cost Map data itself.

MultiCostMapData is a JSON object with each member representing a single Source PID; the name for a member is the PIDName string identifying the corresponding Source PID. For each Source PID, a DstMultiCosts object denotes the associated costs to a set of destination PIDs each identified by a string indexed by PIDName. For each destination PID, object DstMultiCost[PIDName] provides an array of one or several values, each corresponding to the Cost Type listed at the same place in the 'cost-type' array. This array MUST have the same size as the 'cost-type' array. The values in the DstMultiCosts[PIDName] array MUST be listed in the same order as in the 'cost-type' array.

The returned Cost Map MUST include the required Path Costs for each pair of Source and Destination PID for which this information is available. If a cost value is not defined, it MUST be replaced by

the reserved JSON symbol 'null'.

The members 'cost-mode' and 'cost-type' MUST be arrays with the same number of elements.

5.2.6. 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" : {},
  "data" : {
    "cost-mode" : ["numerical", "numerical"],
    "cost-type" : ["routingcost", "hopcount"],
    "map-vtag" : "1266506139",
    "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] }
    }
  }
}
```

5.3. 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.

5.3.1. Media Type

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

5.3.2. HTTP Method

This resource is requested using the HTTP POST method.

5.3.3. 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-type<0..*>;
  CostMode    cost-mode<0..*>;
  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.

cost-mode The Cost Mode for each of the returned Cost Types.
As for the choice of Cost Modes, the ALTO Clients SHOULD be cognizant of operations applicable to different Cost Modes.
For Cost types values encoded with the 'JSONNumber' type, the Cost Mode SHOULD be numerical when the purpose is to perform

multi-variate optimization.

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 (Section XXXX?) 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 (Section XXXX?) 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.

5.3.4. Capabilities

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

```
object {  
  CostMode    cost-modes<0..*>;  
  CostType    cost-types<0..*>;  
  JSONBool    cost-constraints;  
  JSONNumber  max-cost-types; [OPTIONAL]  
} FilteredMultiCostMapCapability;
```

with members:

`cost-modes` See Section 4.2.5 of this MC draft

`cost-types` See Section 4.2.5 of this MC draft

`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.

5.3.5. Response

See Section on Multi Cost Map Service of this draft for the format. The returned Cost Map MUST NOT contain any source/destination pair that was not indicated (implicitly or explicitly) in the input parameters. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as if the PID did not appear in the input parameters. If any constraints are specified, Source/Destination pairs 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.

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

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

```
{
  "meta" : {},
  "data" : {
    "cost-mode" : ["numerical", "numerical"],
    "cost-type" : ["routingcost", "hopcount"],
    "map-vtag" : "1266506139",
    "map" : {
      "PID1": { "PID1": [1,6], "PID2": [5,23], "PID3": [10,5] }
    }
  }
}
```

5.3.7. 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-mode" : ["numerical", "numerical"],
  "cost-type" : ["routingcost", "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" : {},
  "data" : {
    "cost-mode" : ["numerical", "numerical"],
    "cost-type" : ["routingcost", "hopcount"],
    "map-vtag" : "1266506139",
    "map" : {
      "PID1": { "PID2": [5,23], "PID3": [10,5] },
      "PID2": { "PID2": [1,0] },
    }
  }
}
```

5.4. 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.

5.4.1. Media Type

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

5.4.2. HTTP Method

This resource is requested using the HTTP POST method

5.4.3. 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-type<0..*>;
    CostMode      cost-mode<0..*>;
    JSONString    constraints<0..*>;    [OPTIONAL]
    JSONArray     or-constraints<0..*>; [OPTIONAL]
    EndpointFilter endpoints;
} ReqEndpointMultiCostMap;
```

with members:

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

`cost-type` Defined equivalently to the "cost-type" 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.

5.4.4. Capabilities

See section on Filtered Multi Cost Map capabilities in this draft.

5.4.5. Response

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

```
object EndpointDstMultiCosts {
  JSONArray [TypedEndpointAddr];
  ...
};

object {
  EndpointDstMultiCosts [TypedEndpointAddr]<0..*>;
  ...
} EndpointMultiCostMapData;

object {
  CostMode          cost-mode<0..*>;
  CostType          cost-type<0..*>;
  EndpointMultiCostMapData map;
} InfoResourceEndpointMultiCostMap;
```

InfoResourceEndpointMultiCostMap has members:

cost-type<0..*> The Cost Types used in the returned Cost Map.

cost-mode<0..*> The Cost Mode for each of the Cost Types used
in the returned Cost Map.

map The Endpoint Multi-Cost Map data itself.

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

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

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

```
{
  "meta" : {},
  "data" : {
    "cost-type" : ["routingcost", "hopcount"],
    "cost-mode" : ["numerical", "numerical"],
    "map" : {
      "ipv4:192.0.2.2": {
        "ipv4:192.0.2.89" : [1, 7],
        "ipv4:198.51.100.34" : [2, 4],
        "ipv4:203.0.113.45" : [3, 2]
      }
    }
  }
}
```

6. 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],

6.1. 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.

6.2. 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.

7. Acknowledgements

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

Sabine Randriamasy is partially supported by the MEDIEVAL project (<http://www.ict-medieval.eu/>), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the MEDIEVAL project or the European Commission.

8. References

8.1. 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.

8.2. Informative References

- [ID-ALTO-Requirements13]
"draft-ietf-alto-reqs-13.txt", January 2012.
- [ID-alto-protocol]
, Eds., "ALTO Protocol" draft-ietf-alto-protocol-10.txt",
October 2011.
- [ID-alto-protocol-11]
, Eds., "ALTO Protocol" draft-ietf-alto-protocol-11.txt",
March 2012.
- [draft-jenkins-alto-cdn-use-cases-01]
"Use Cases for ALTO within CDNs"
draft-jenkins-alto-cdn-use-cases-01", June 2011.
- [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 (editor)
Alcatel-Lucent Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

W. D. Roome
Alcatel-Lucent Bell Labs
600 Mountain Ave.
Murray Hill, NJ 07974
USA

Phone:
Fax:
Email: W.Roome@alcatel-lucent.com
URI:

Nico Schwan

Email: ietf@nico-schwan.de

ALTO
Internet-Draft
Intended status: Standards Track
Expires: April 18, 2013

D. Wang
H. Song
N. Zong
Huawei
Oct 15, 2012

ALTO and DECADE Integration
draft-wang-alto-decade-integration-00

Abstract

While DECADE architecture is going to provide a framework to use in-network storage with standard interface for content distribution, there are still cases when many in-network storage servers provide the same content. Then traffic optimization mechanism is needed to select one best replica for the user request. This document is going to discuss the integration of Application Layer Traffic Optimization (ALTO) protocol into DECADE so as to optimize network traffic.

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 April 18, 2013.

Copyright Notice

Copyright (c) 2012 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. Terminology | 3 |
| 3. Integration Architecture | 3 |
| 3.1. Function Entity | 3 |
| 3.2. Integration Architecture | 3 |
| 3.3. Process Flow | 5 |
| 4. Use cases | 6 |
| 4.1. P2P File Sharing | 6 |
| 4.2. Offline P2P live streaming | 7 |
| 5. ALTO Protocol Considerations | 9 |
| 6. Security Considerations | 10 |
| 7. IANA Considerations | 11 |
| 8. References | 11 |
| 8.1. Normative References | 11 |
| 8.2. Informative Reference | 11 |
| Authors' Addresses | 11 |

1. Introduction

In-network storage is popular for content distribution, but existing framework either keeps applications out of the control loop or need to support variety of application protocol and implementations. DECADE architecture provides the general content distribution framework with standard interface for resource control and data transport. But either using the DECADE compatible in-network storage servers as personal storage or using it as content distribution platform for applications, there are possibilities that multiple servers provide the same content, which would a good scenario to consider using Application Traffic Optimization (ALTO) Protocol to select a better-than-random server.

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 RFC-2119 [RFC2119]..

This document reuses the terminology defined in [I-D.draft-alto-problem-statement] and [I-D.draft-decade-problem-statement].

3. Integration Architecture

3.1. Function Entity

ALTO Server: The logical entity that provides interfaces to the queries to the ALTO service [RFC5693].

DECADE Server: A DECADE Server stores DECADE data inside the network, and thereafter manages both the stored data and access to that data [I-D.ietf-decade-arch-02].

DECADE Client: A DECADE Client stores and retrieves data at DECADE Servers [I-D.ietf-decade-arch-02].

App Tracker: Index Server that stores current peer list which is sharing certain network resource, such as movies or music, and so on.

3.2. Integration Architecture

Figure 1 shows the basic integration architecture framework of integrating DECADE and ALTO. It indicates that integrating DECADE and ALTO can provide optimized network traffic by selecting serving

peers in a better-than-random model.

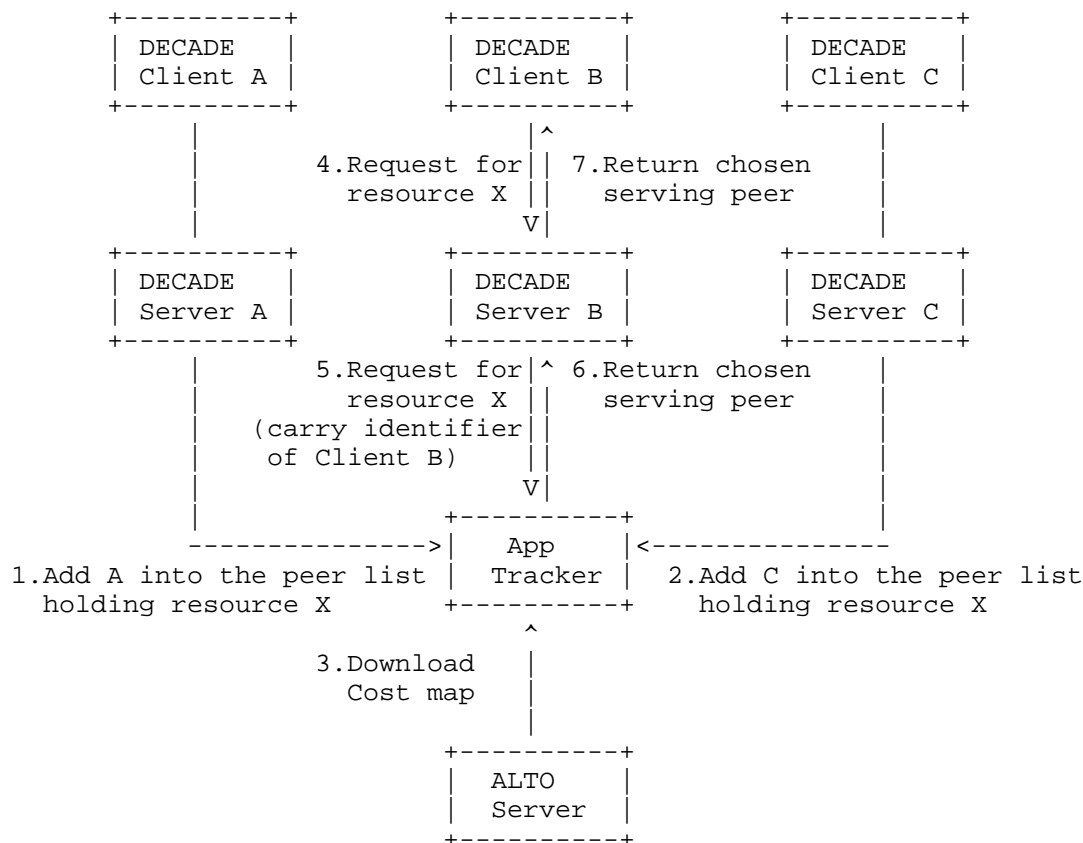


Figure 1 ALTO and DECADE Integration Architecture Framework

In this framework, DECADE Client requests certain network resources from its in-network storage (DECADE Server). If its DECADE Server contains such resources, the request would be satisfied immediately and DECADE Client can download the resource directly. Otherwise, its DECADE Server will query the App Tracker for the best serving peer which contains those resources, carrying the identifier of the requesting DECADE Client in the request. Each time when App Tracker receives DECADE Server's request for certain resources, it queries for all current serving peers that contain such resources. It then looks up the network map it already downloaded from the ALTO Server in advance. The cost between the requesting DECADE Client and each serving peer in the peer list will be calculated. Finally, the best serving replica at the lowest cost will be returned and the requesting resources can be downloaded from it. This scheme embodies the idea of integration of DECADE and ALTO, which may bring many

benefits, such as saving network bandwidth.

In this framework, the cost between every two serving peers can be looked up from ALTO Server, however, the cost between a DECADE Client and any other Client's DECADE Server cannot be obtained. Therefore, it'd better to pre-store network map with cost between each DECADE Client and its DECADE Server in App Tracker. In this way, combining with cost map provided by ALTO Server, the cost between a DECADE Client and any other Client's DECADE Server can be calculated, and the lowest cost can be decided.

3.3. Process Flow

Figure 2 shows the process flow of integrating DECADE and ALTO .

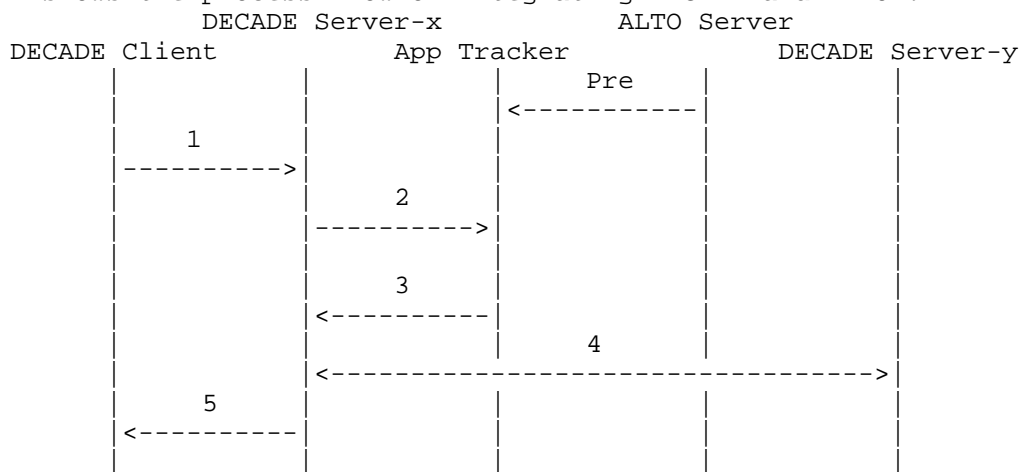


Figure 2 Process Flow of Integrating DECADE and ALTO

Pre: App Tracker downloads cost map from ALTO Server in advance. This step can happens any time before step 3.

1. DECADE Client requests for certain network resources from its DECADE Server (here we name it DECADE Server-x).

2. If DECADE Server-x doesn't include the requested resources, it will ask for App Tracker's suggestion. The identifier of DECADE Client will be sent to the App Tracker.

3. APP Tracker looks up all current serving peers that contain such resources. It then calculate the cost between the requesting Client and each serving peers based on the cost map it downloaded from ALTO Server in advance and the pre-stored network map with cost between each DECADE Client and its DECADE Server. Finally, the most suitable serving peer (here we name it DECADE Server-y) at the lowest cost

will be picked up and returned to DECADE Server-x.

4. DECADE Server-x downloads the data object from DECADE Server-y.

5. DECADE Client downloads the data object from its DECADE Server (DECADE Server-x).

4. Use cases

4.1. P2P File Sharing

Related Applications: There are many P2P file sharing applications, e.g., BitTorrent, Shareaza, Ares. The most widely used is BitTorrent.

Original setting: In BitTorrent without DECADE and ALTO, a client uploads to other clients using its last mile. This consumes expensive last-mile uplink bandwidth.

BitTorrent with DECADE: A client first uploads files to a DECADE Server, who can then serve other clients.

BitTorrent with DECADE and ALTO: While two DECADE Servers hold the same file that a client wants, integration DECADE with ALTO can help the client decide which DECADE Server is the best replica to get this file. And in this scheme, it may bring many benefits, such as saving network bandwidth.

Figure 3 shows an example. In this example, DECADE Client B wants to get a file which both DECADE Server A and DECADE Server C hold. In the original setting without integrating DECADE with ALTO, DECADE Client B could randomly get the file by its DECADE Server (DECADE Server B) from either DECADE Server A or DECADE Server C. In this case, DECADE Client B would consult App Server which DECADE Server is the most suitable serving peer. In this process, ALTO Server provides cost map, and the network map with cost between each DECADE Client and its DECADE Server is pre-stored in App Tracker. Based on those two maps, the cost between each serving DECADE Server and the requesting DECADE Client can be calculated, finally the serving peer at the lowest cost would be selected.

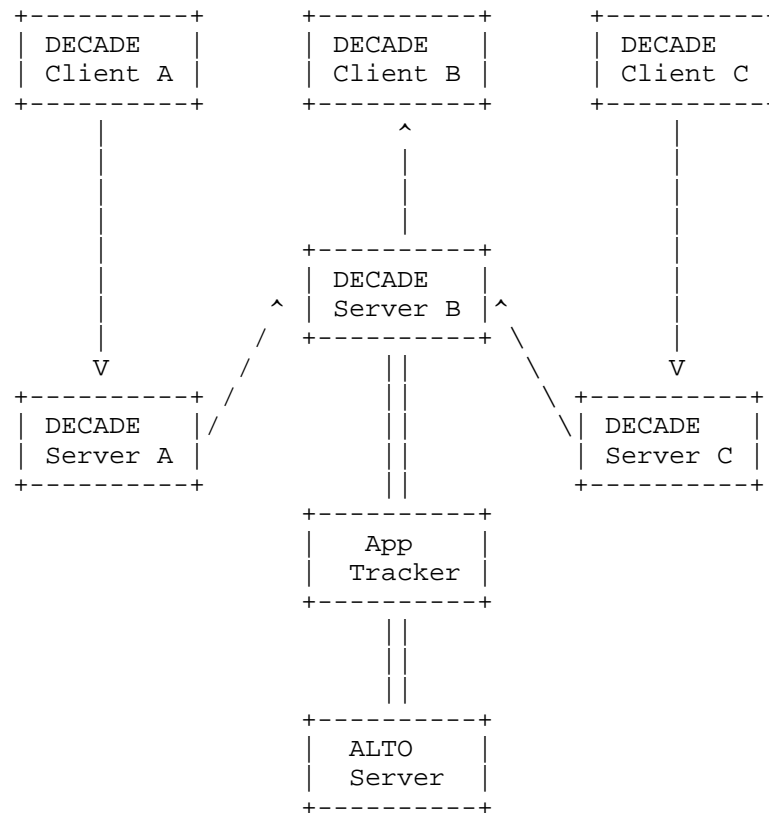


Figure 3 P2P File Sharing

Benefits: DECADE Client can get file by its DECADE Server from a serving peer at the lowest cost, which may saving traffic. Of course, last mile uplink bandwidth is saved, which is expensive.

4.2. Offline P2P live streaming

Related Applications: There are many P2P live streaming applications, e.g., PPLive, PPStream.

Original setting: In P2P live streaming without DECADE and ALTO, clients may get a bad user experience when they watch the streaming after broadcasting at a later time, since there will be not enough audiences to watch it after living broadcast.

PPLive with DECADE: By using DECADE, a client can watch a live streaming at a later time, and still have a good video quality.

PPLive with DECADE and ALTO: While a client wants to watch a live

streaming at a later time, integrating DECADE and ALTO can help it decide a DECADE Server which hold the content and also at the lowest cost. The client can also enjoy a good user experience, instead of having a bad video quality.

Figure 4 shows an example. In this example, Client B is broadcasting a live streaming. Client A is online and watching the live streaming, and DECADE Server A downloads this streaming from DECADE Server B in real time. Client C is offline and wants to watch this streaming at a later time. DECADE Server C can download this streaming from DECADE Server A and DECADE Server B after comparing their cost by consulting App Tracker and ALTO Server. Then DECADE Server C will hold the living streaming data for Client C. After Client C once is online, it can get the streaming directly for its DECADE Server C.

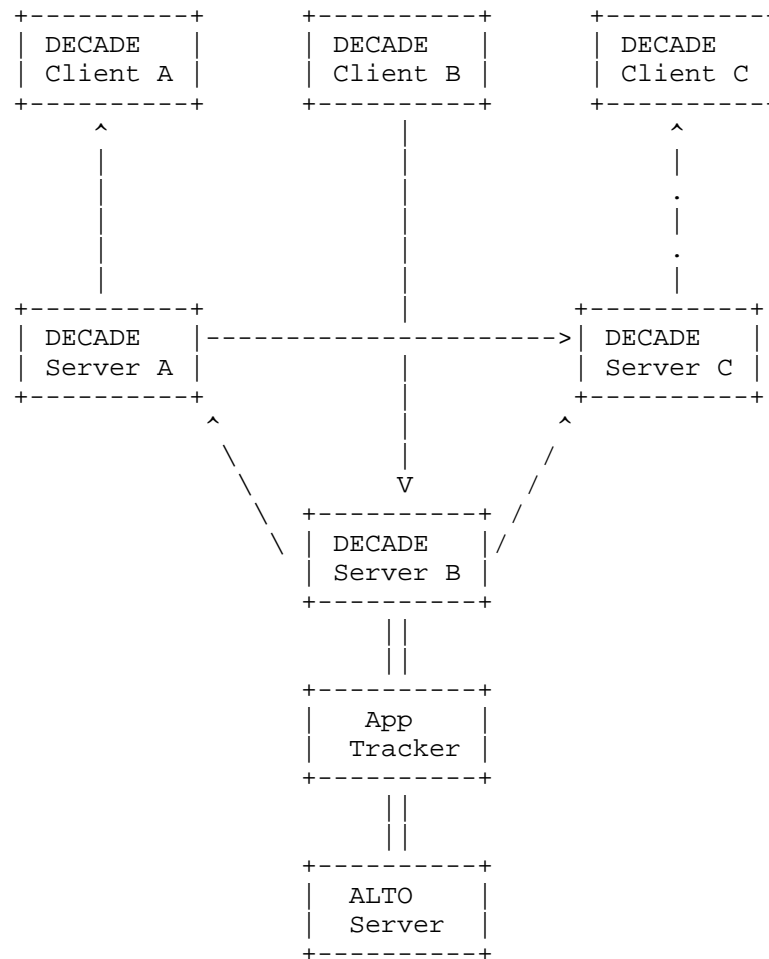


Figure 4 Offline P2P Live Streaming

Benefits: In original setting, clients missed live streaming broadcast, may get a bad video quality when they watch it at a later time. While in integrating DECADE with ALTO, the video quality is also good since Client C downloads it directly from its DECADE Server. Also, the new setting can help save network traffic as well as last mile uplink bandwidth in this use case.

5. ALTO Protocol Considerations

In this context, App Tracker may currently need a variety of information to perform serving peer selection, for example, ALTO Cost Map from ALTO Server. Via embedding ALTO Client in App Tracker, App

Tracker can obtain and locally store ALTO Cost Map from ALTO Server by ALTO Protocol, and benefit from this ALTO information to enable network-efficient traffic patterns and improve serving performance. Figure 5 shows the interactions among the main entities in our integration framework where App Tracker is an ALTO Client and applies cost map when selecting the best serving peer.

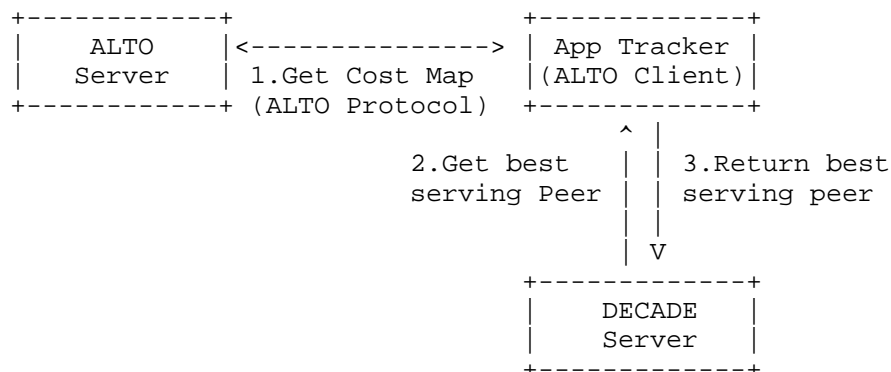


Figure 5 ALTO Client Embedded in App Tracker

1. The App Tracker requests the Cost Map amongst all PIDs from the ALTO Server by ALTO Protocol.
2. A DECADE Server requests current serving peer at lowest cost from the App Tracker.
3. The App Tracker returns current best serving peer, which is computed based on the ALTO Cost Map as well as other information pre-stored in the App Tracker.

Since App Tracker acts as the ALTO Client in our scheme, while applying ALTO Protocol, "Endpoint" in the protocol is used to carry/define information of the App Tracker, 'including address type, address of App Tracker, et, al. ALTO Cost Map is obtained from ALTO Server via an HTTP GET method without any input parameters. The media type is "application/alto-costmap+json". And ALTO Server would return cost map or an error code to the App Tracker after receiving such request.

6. Security Considerations

This document does not consider security issues. Security will be considered in further version of this draft.

7. IANA Considerations

There is no IANA consideration for this document.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.
- [RFC5693] Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", October 2009.

8.2. Informative Reference

- [I-D.ietf-decade-arch]
Alimi , R., Yang, Y., Rahman, A., Kutscher, D., and H. Liu, "DECADE Architecture", July 2011.
- [I-D.ietf-decade-problem-statement]
Song, H., Zong, N., Yang, Y., and R. Alimi, "DECoupled Application Data Enroute (DECADE) Problem Statement".
- [I-D.draft-alto-problem-statement]
Seedorf, J. and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement", March 2009.

Authors' Addresses

Danhua Wang
Huawei

Email: wangdanhua@huawei.com

Haibin Song
Huawei

Email: haibin.song@huawei.com

Ning Zong
Huawei

Email: zongning@huawei.com

