ALTO WG
Internet-Draft
Intended status: Standards Track
Expires: January 5, 2015

M. Scharf
Alcatel-Lucent Bell Labs
July 4, 2014

ALTO Extension for YANG Modules
draft-scharf-alto-yang-00

Abstract

The Application-Layer Traffic Optimization (ALTO) protocol is designed to expose network topology information to applications. The ALTO protocol includes a well-defined set of base services. This document specifies an optional ALTO extension that enables ALTO clients to discover and query additional data being defined by YANG modules. This document describes the corresponding extensions of the ALTO Information Resource Directory and other required mechanisms.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Application-Layer Traffic Optimization (ALTO) protocol [I-D.ietf-alto-protocol] is designed to expose network topology information to applications. The ALTO Information Resource Directory (IRD) of an ALTO server describes the services supported by an ALTO server and corresponding attributes. As defined in [I-D.ietf-alto-protocol], each information resource has several associated attributes, including an assigned identifier, a response format, capabilities, accepted input parameters, and other resources that it may depend on. Prior to obtaining ALTO guidance, ALTO client may retrieve the Information Resource Directory from the ALTO server to determine the attributes of individual information resources that this ALTO Server provides.

The ALTO protocol is designed to be extensible, and several protocols extensions have been suggested, including for instance [I-D.roome-alto-pid-properties] or [I-D.randriamasy-alto-multi-cost]. In addition to such extensions of the ALTO protocol, an ALTO server may be willing to offer additional network information and guidance beyond the base functionality of the ALTO protocol. In this case, one option is to specify the offered guidance using YANG [RFC6020] as data modeling language.

This document specifies an optional ALTO extension that enables ALTO clients to discover and query additional data offered by an ALTO server, being defined by YANG modules. This document describes the corresponding extensions of the ALTO Information Resource Directory and other required mechanisms.

2. Terminology

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

3. ALTO Network Topology Exposure vs. Network Configuration

ALTO is designed as a protocol between clients integrated in applications and servers that provide network information and guidance (e.g., basic network location structure and preferences of network paths). The objective is to modify network resource consumption patterns at application level while maintaining or improving application performance. The basic information of ALTO is based on an abstract representation of the network as it matters to applications at endpoints.

Given the focus on topology exposure to guidance, there are several differences between ALTO and other protocols used for network management, such as SNMP [RFC3411] or NETCONF [RFC6241]:

- Specialization: ALTO is an optimized protocol focused on exposing network topology data. This is different to general-purpose protocols.

- Read-only: ALTO is a query/response protocol to retrieve data. Neither ALTO network/cost map queries nor queries to the ALTO endpoint property or cost services are designed to affect state in the ALTO server or in the ALTO client. Network management protocols are typically designed to manipulate the configuration of network devices.

- Trust model: The ALTO protocol has been designed for use cases where the ALTO server and client can be located in different organizations or trust domains. Network configuration will require a much stronger security model.

- Non-NMS applications: ALTO is designed for applications that perform resource selections among endpoints, e.g., in an application overlay. Such applications are typically not part of an Element Management System (EMS) or Network Management System (NMS).

- A priori knowledge: ALTO does not assume that an ALTO client has any a priori knowledge about the ALTO server and its supported features. An ALTO server can be discovered automatically. In network management, there is often a need for explicit configuration, e.g., of security credentials.

- Terminology: In ALTO, the application is running a client, while the network offers a service by means of a server. SNMP would use the terms manager and agent for these entities, and NETCONF may use the terms server and client in the opposite way. This reflects the different usage patterns.

These aspects are also detailed in [I-D.ietf-alto-deployments].

The same differences also apply to control plane protocols designed to configure state on network elements, such as components of a Path Computation Element (PCE) architecture or an Interface to the Routing System (I2RS).

4. Benefits of Using YANG Data Models

Despite different use cases, ALTO clients could use a limited set of functions originally designed for network management, including protocol functions for read-only retrieval of data. For instance, an ALTO server may use network monitoring data or Operations, Administration, and Maintenance (OAM) measurement results as data sources [I-D.ietf-alto-deployments], and that data could originate from Network Management Systems (NMS). If permitted by privacy policies, an ALTO server could be willing to optionally expose such input data used by the ALTO service to ALTO clients.

In that case, a straightforward solution is to transform that input data into a format supported by ALTO, e.g., network and cost maps with additional cost metrics. Yet, if the ALTO server already queries data specified in YANG [RFC6020], it could optionally also expose that data to ALTO clients, provided that the aforementioned restrictions are taken into account.

Support of YANG modules would be a simple way for an ALTO server to offer optional, additional services, e.g., to access monitoring data. A data model described in YANG could enable an ALTO client to easily determine the type of additional data exposed by ALTO, since a YANG module specifies the corresponding syntax.

As a specific example, an ALTO server may have access to interface statistics for endpoints in the network, using the YANG module "ietf-interfaces" [RFC7223], and it may use these statistics to e.g. to calculate network and cost maps. If an ALTO client would

specifically be interested in counter metrics such as "out-errors", the ALTO server could obviously expose this by ALTO, e.g., as several cost maps or as an endpoint/PID property [I-D.roome-alto-pid-properties]. An alternative approach would be that the ALTO server natively exposes that data in a YANG-based format. This requires a solution to support YANG modules in ALTO.

5. Extending ALTO to Support YANG Modules

This section specifies how an ALTO server can announce the support of additional, optional services defined by YANG modules.

5.1. ALTO Information Resource Directory

According [I-D.ietf-alto-protocol], objects in the ALTO Information Resource Directory (IRD) have the following format:

```
object {
  JSONString      uri;
  JSONString      media-type;
  [JSONString     accepts;]
  [Capabilities   capabilities;]
  [ResourceID     uses<0..*>;]
} IRDResourceEntry;
```

The extension in this document use this format and defines the following setting of these objects:

- uri: A URI at which the data is provided. This can both be an URI at the ALTO server, or it can delegate to another server (Section 9.2.4 of [I-D.ietf-alto-protocol]). This extension uses the URI to refer to data described in YANG modules.

- media-type: The media type that is to be used by GET (or POST) requests to the corresponding URI. This is further detailed below.

- accepts: This optional parameter can also be set if required, as described in [I-D.ietf-alto-protocol].

- capabilities: Capabilities are optional. A server MAY announce the use of YANG as data modeling language for certain resources as a capability ("yang" : true), if this is not already implied by the media-type. Further capabilities might be used to describe the YANG modules, but this is TBD.

- uses: The setting of this parameter is for further study. According to [I-D.ietf-alto-protocol], this setting can be used to

define other resources on which this resource directly depends. As a result, it could for instance be used to refer to ALTO cost maps that are related to the data described in this resource refering to YANG modules.

An example of a corresponding ALTO Information Resource Directory entry would be:

```
...
"my-own-map-data" : {
  "uri" :
    "http://alto.example.com/restconf/data/my-own-map-data",
  "media-type" : "application/yang.data+json",
  "capabilities" : {
    "yang" : true
  }
},
...
```

5.2. JSON Encoding

The ALTO protocol uses a REST-ful design and encodes its requests and responses using JSON [RFC4627]. Therefore, all data returned using this specification MUST be encoded in JSON.

[I-D.ietf-netmod-yang-json] defines rules for representing data defined using YANG as JSON text.

5.3. YANG Data Model Assumptions

YANG can model configuration data as well as state data. The latter is marked by the "config false" statement [RFC6020]. State data on a system is no configuration data and cannot be changed, such as read-only status information and collected statistics. When a node in a YANG module is tagged with "config false", its sub-hierarchy is flagged as state data as well.

Since ALTO only offers read-only access to information, this document implicitly assumes that the URI announced by an ALTO server only exposes read-only data. This document only describes how an ALTO client can discover the URI for additional data and learn now to retrieve data by a REST protocol from there; the actual communication and interpretation of data is outside the scope of this document.

Read-only access can trivially be enforced if an ALTO server using the mechanism in this document only use YANG modules that contain state data only, i.e., all containers, lists, and leafs etc. are be covered by "config false" statements. Alternatively, there could be

ways to ensure read-only access, e.g., by appropriate selection of the URI.

Any write access by a client using the information in the ALTO IRD is outside the scope of this memo.

This document does not cover notifications. The handling of "notifications" in a YANG module is therefore outside the scope of the document.

The handling of "rpc" statements in a YANG module is for further study.

5.4. Media Type

This document does not register additional media types. An ALTO client SHOULD use the media type(s) defined in the ALTO IRD to access the URI, as detailed in the next section.

An example media type could be "application/yang.data+json" as defined in [I-D.ietf-netconf-restconf].

5.5. Accessing the URI

The way to retrieve data from the URI returned by the ALTO IRD depends on the media type. An ALTO client can determine from the media type whether it supports the method and whether optional data from an ALTO server can be retrieved. This document describes an optional ALTO extension and does not mandate that an ALTO client supports any specific protocol.

One approach to retrieve additional data described by a YANG model would be RESTCONF [I-D.ietf-netconf-restconf]. In this case, the ALTO IRD is basically used as a discovery mechanism for corresponding RESTCONF URIs. For RESTCONF, the syntax of the URI, as well as the protocol for data retrieval is defined in [I-D.ietf-netconf-restconf]. For instance, a valid URI would be "http://alto.example.com/restconf/data/my-own-map-data", and a corresponding media type could be "application/yang.api". If the URI points to the RESTCONF root, the client could also use other APIs as defined in [I-D.ietf-netconf-restconf], e.g., to retrieve the YANG module definition file. The latter would enable an ALTO client to dynamically learn the YANG module definitions.

In a simpler variant, an ALTO server would just return the complete tree of state data encoded in JSON [I-D.ietf-netmod-yang-json]. This could be sufficient for extensions equivalent to the current ALTO network and cost map services without filtering, which always return

all available data without fine-grained query access. Details of this mode are for further study.

6. Example

In the following, we illustrate the mechanism described in this document with a simple example.

The initial query of an ALTO client to an ALTO server's IRD could be:

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

The server would then return the supported ALTO services, including the one specified in this document:

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

{
  "meta" : {
     ...
  },
  "resources" : {
     ...
     "my-own-map-data" : {
        "uri" :
          "http://alto.example.com/restconf/data/my-own-map-data",
        "media-type" : "application/yang.data+json",
        "capabilities" : {
          "yang" : true
        }
      },
     ...
  }
}
```

Since the media type refers to [I-D.ietf-netconf-restconf], a follow-up query of the ALTO client supporting this protocol could be:

```
GET /restconf/data/my-own-map-data  HTTP/1.1
Host: alto.example.com
Accept: application/yang.data+json
```

The response of the server is outside the scope of this document and depends on the YANG data model accessed by that URI. For instance, assume the following trivial model of an extension just giving a convenient list of the PIDs currently used by the ALTO server:

```
module my-own-map-data {
  ...
  container my-own-map-data {
    config false;
    list pid-list {
      key pid;
      leaf pid {
        type string;
      }
    }
  }
  ...
}
```

Then, the server response could be:

```
HTTP/1.1 200 OK
Content-Length: TBD
Content-Type: application/yang.data+json

{
  "my-own-map-data" : {
    "pid-list" : [
      {
        "pid" : "PID1",
      },
      {
        "pid" : "PID2",
      },
      ...
      {
        "pid" : "PID9",
      }
    ]
  }
}
```

This is a trivial example, but the same principle is applicable to any YANG module. Obviously, an ALTO client has to support processing of the corresponding YANG module (or modules) to make use of this extension.

7. IANA Considerations

TBD

8. Security Considerations

This document descibes read-only access to data optionally announced by an ALTO server. The security considerations of the ALTO protocol as detailed in [I-D.ietf-alto-protocol] and [I-D.ietf-alto-deployments] apply to the ALTO extension in this document. Specifically, the operator of an ALTO server has to ensure that there is no unauthorized access to sensitive data. This extension should not be used if there are privacy concerns.

9. Conclusion

TBD

10. References

10.1. Normative References

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

[I-D.ietf-netconf-restconf]
Bierman, A., Bjorklund, M., Watsen, K., and R. Fernando, "RESTCONF Protocol", draft-ietf-netconf-restconf-01 (work in progress), July 2014.

[I-D.ietf-netmod-yang-json]
Lhotka, L., "JSON Encoding of Data Modeled with YANG", draft-ietf-netmod-yang-json-00 (work in progress), April 2014.

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

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

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

10.2. Informative References

[I-D.ietf-alto-deployments]
Stiemerling, M., Kiesel, S., Previdi, S., and M. Scharf, "ALTO Deployment Considerations", draft-ietf-alto-deployments-10 (work in progress), July 2014.

[I-D.randriamasy-alto-multi-cost]
Randriamasy, S., Roome, B., and N. Schwan, "Multi-Cost ALTO", draft-randriamasy-alto-multi-cost-07 (work in progress), October 2012.

[I-D.roome-alto-pid-properties]
Roome, W. and Y. Yang, "PID Property Extension for ALTO Protocol", draft-roome-alto-pid-properties-02 (work in progress), July 2014.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, May 2014.

Author's Address

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

Email: michael.scharf@alcatel-lucent.com