Yang Data Model for OAM and Management of ALTO protocol

draft-ietf-alto-oam-yang

Jingxuan Zhang
Dhruv Dhody
Roland Schott
Kai Gao

ALTO WG @ IETF 115

Current Status

Main goal: Define a YANG data model for Operations, Administration, and Maintenance (OAM) & Management of ALTO Protocol.

Latest version: https://datatracker.ietf.org/doc/html/draft-ietf-alto-oam-yang-02

Editor's copy on GitHub:
https://ietf-wg-alto.github.io/draft-ietf-alto-oam-yang/draft-ietf-alto-oam-yang.html

YANG modules:
https://github.com/ietf-wg-alto/draft-ietf-alto-oam-yang/tree/main/yang

Current Status (Cont.)

Open discussions and progress since IETT 114:

- T1: How to handle data types defined by IANA registries (e.g., ALTO cost modes and metrics)
  - https://mailarchive.ietf.org/arch/msg/alto/S10Ua4tvVhPGu6FFJhbhDGBbPXs/
- T2: Whether and how to supply server-to-server communication for multi-domain settings
  - https://mailarchive.ietf.org/arch/msg/alto/MvVDgeZnmi-_0sY8al0hGuWbWU8/
- T3: How to build connection between data sources and algorithm data model
  - https://datatracker.ietf.org/doc/html/draft-ietf-alto-oam-yang-02#appendix-A

Achieved Milestones:

- IETF 115 Hackathon
  - Partial implemented ALTO O&M data model in OpenALTO implementation: https://github.com/openalto/alto

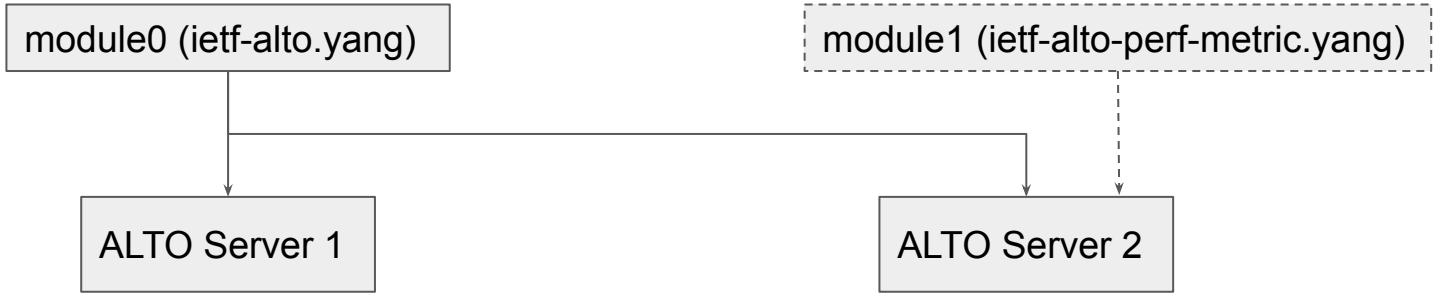
Overall Update: Reorganize the Contents

Exactly align with all the 7 basic requirements in order.

Put implementation-specific data model as an example in appendix.

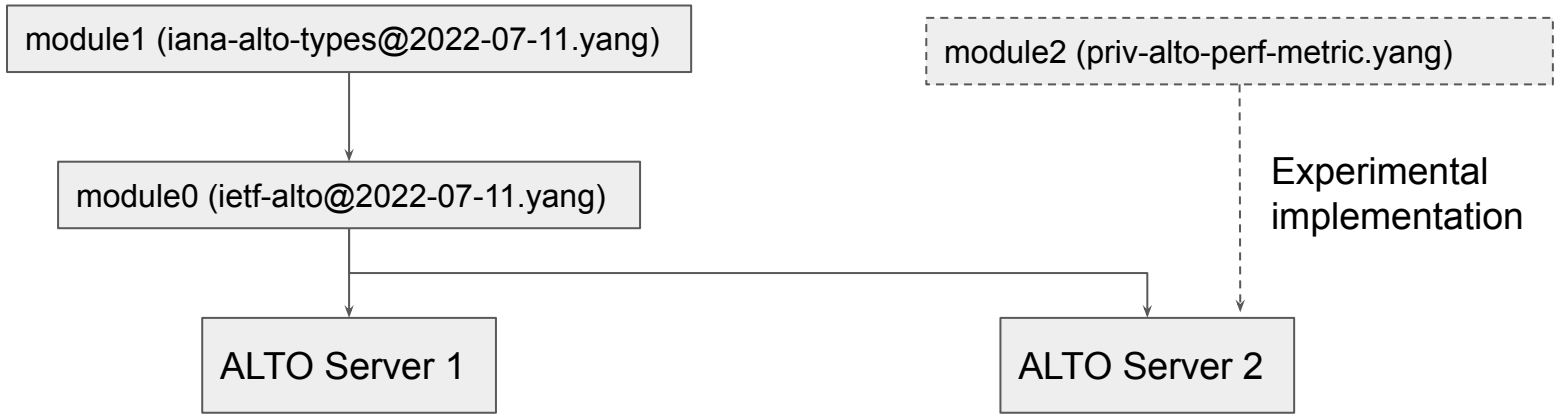
T1: Data Types in ALTO Related IANA Registries

- The decision is to use identity to define data types.
  - This allows the data types to be managed in a more modular way, and to guarantee backward compatibility.
  - Future documents can define new data types by adding new identities in extension modules. Once a document becomes a standard, the new extension module will also be added to the standard IETF YANG module base.



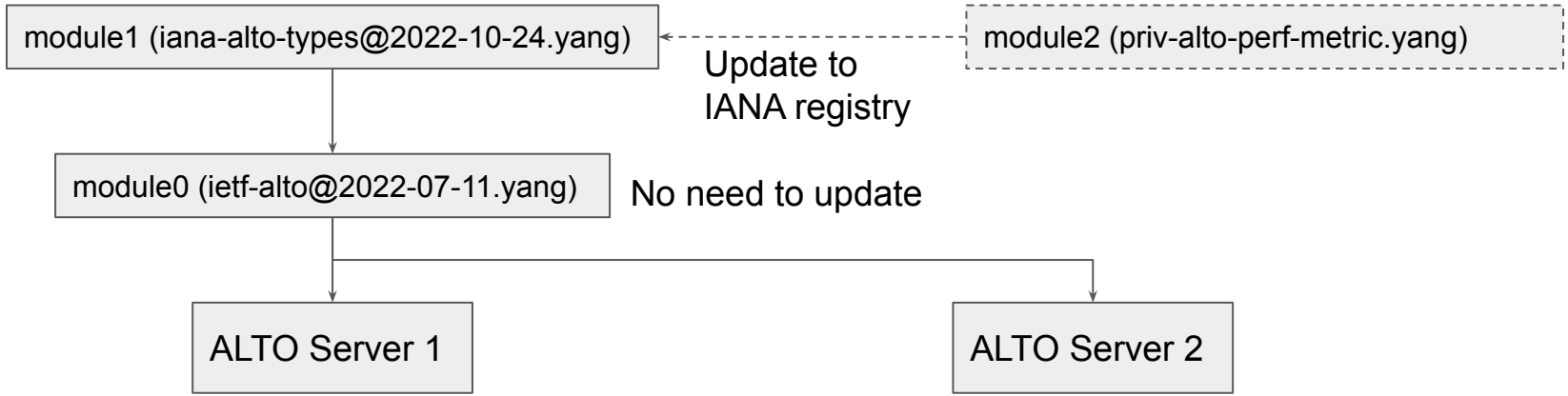
T1: Data Types in ALTO Related IANA Registries (Cont.)

- WG has different opinions about whether use an IANA-maintained module (e.g., iana-alto-types.yang):
  - Support: IANA-maintained module can guarantee compatibility
  - Not support: If not expect to have frequent changes, IANA-maintained module is overdesign



T1: Data Types in ALTO Related IANA Registries (Cont.)

- WG has different opinions about whether use an IANA-maintained module (e.g., iana-alto-types.yang):
  - Support: IANA-maintained module can guarantee compatibility
  - Not support: If not expect to have frequent changes, IANA-maintained module is overdesign



T2: Server-to-Server Communication

O&M data model to configure server-to-server communication requires the following functionalities:

- **T2.1**: Configure how the server to **be discovered by another server/client**
  - **Status: ready**
- T2.2: Configure how the server to **discover another server**
  - **Status: in progress**
- T2.3: Configure how the server to **communicate to a discovered server**
  - **Status: not determined whether should be in the scope**

```
grouping alto-server-discovery-grouping:
  +-- (server-discovery-manner)?
     +--:(reverse-dns)
     |  +-- rdns-naptr-records
     |     +-- static-prefix*           inet:ip-prefix
     |     +-- dynamic-prefix-source*
     |              -> /alto-server/data-source/source-id
     +--:(internet-routing-registry)
     |  +-- irr-params
     |     +-- aut-num?   inet:as-number
     +--:(peeringdb)
        +-- peeringdb-params
           +-- org-id?   uint32
```

For T2.1, the current data model provide predefined cases for server discovery learned from practical deployment but can be extended through augmentation:

- Reverse DNS: RFC8686
- IRR: RFC2622
- PeeringDB: https://www.peeringdb.com/

T2: Server-to-Server Communication

O&M data model to configure server-to-server communication requires the following functionalities:

- T2.1: Configure how the server to **be discovered by another server/client**
  - **Status: ready**
- **T2.2**: Configure how the server to **discover another server**
  - **Status: in progress**
- T2.3: Configure how the server to **communicate to a discovered server**
  - **Status: not determined whether should be in the scope**

```
grouping alto-server-discovery-client-grouping
  +-- (server-discovery-client-manner)?
     +--:(rdns)
     |  +-- rdns-params
     |     +-- dns-server          inet:host
     +--:(internet-routing-registry)
     |  +-- irr-params
     |     +-- whois-server        inet:host
     +--:(peeringdb)
        +-- peeringdb-params
           +--peeringdb-endpoint  inet:uri
```

Similar to T2.1, T2.2 provides common model for how to access an existing server discovery system.

Predefined server discovery systems are aligned with mechanism defined in `alto-server-discovery-grouping`.

The model can also be extended by augmentation.

T2: Server-to-Server Communication

O&M data model to configure server-to-server communication requires the following functionalities:

- T2.1: Configure how the server to **be discovered by another server/client**
  - **Status: ready**
- T2.2: Configure how the server to **discover another server**
  - **Status: in progress**
- **T2.3**: Configure how the server to **communicate to a discovered server**
  - **Status: not determined whether should be in the scope**

There are multiple potential solutions:

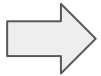
- C/S mode using ALTO
- C/S mode using other underlay protocols
- Peering mode using other underlay protocols

None of them has become the standard yet.

Implementation & Deployment updates will discuss more details.

C/S mode using ALTO can be the simplest approach to leverage existing ALTO standards.

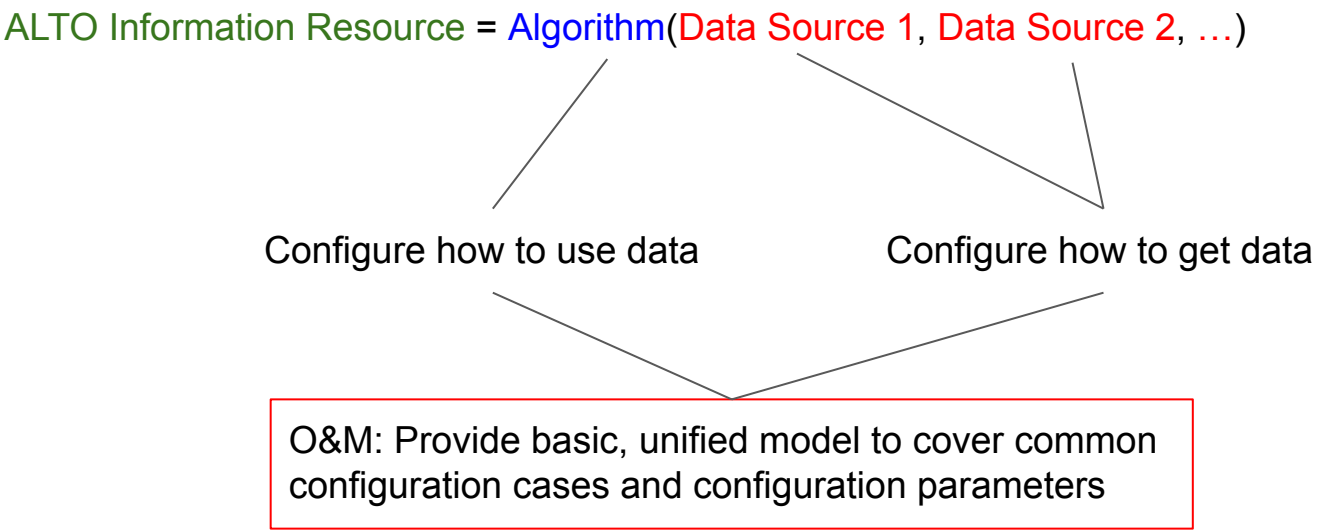
```
module: ietf-alto
  +--rw alto-server
     +...
```



```
module: ietf-alto
  +--rw alto-server
  |  +...
  +--rw alto-client* [id]
     +...
```

T3: Connection between Data Sources and Algorithms

As suggested by RFC7285 (Sec 16.2.4), data sources and algorithms are two major components to be configured.



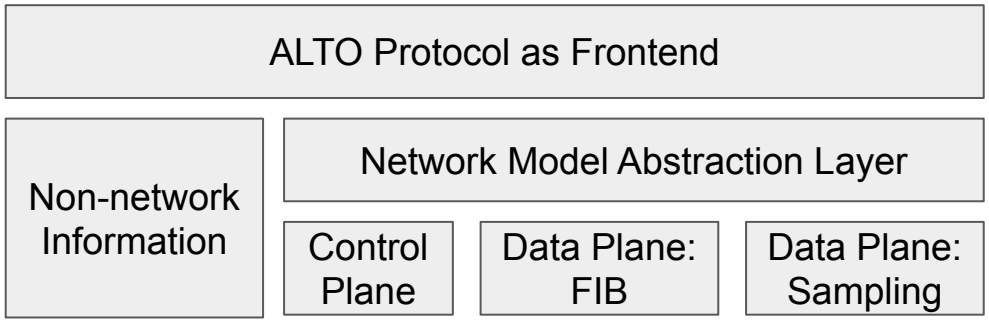
T3: Connection between Data Sources and Algorithms (cont.)

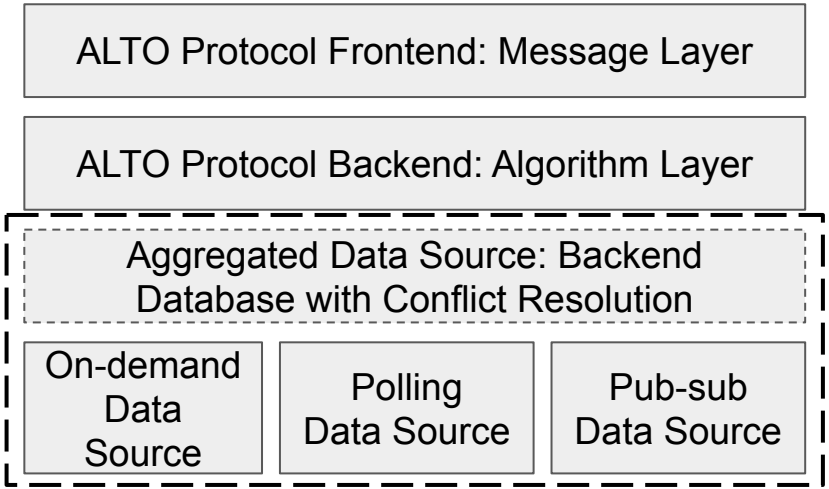
From implementation & deployment perspective:

- How to handle heterogeneous formats of data sources
- How to process data collected from data sources

From O&M perspective:

- How to handle heterogeneous mechanisms to access data sources
- How to correctly configure calling flows for information resource creation



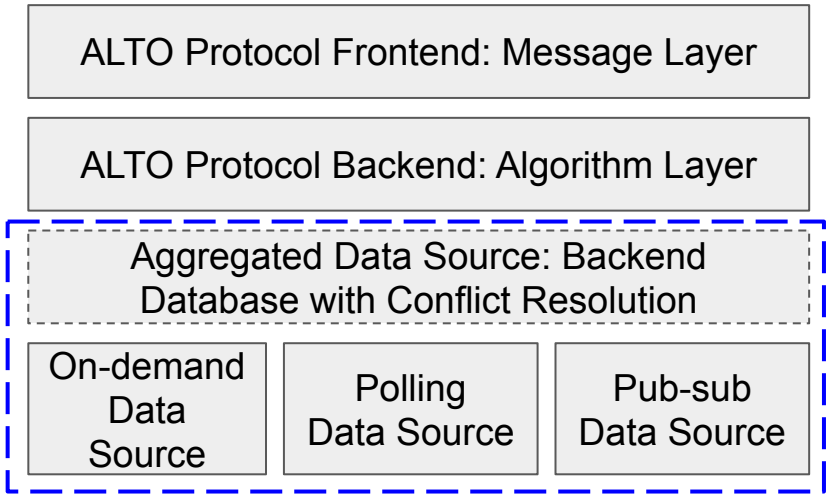


A real deployment in https://alto.nrp-nautilus.io/directory/default (IRD):

```
λ kubectl get deployments | grep openalto
openalto-agent      1/1     1            1           2d23h
openalto-db         1/1     1            1           2d23h
openalto-frontend   1/1     1            1           2d23h
```

T3: Connection between Data Sources and Algorithms (cont.)

Mapping from O&M perspective to data model:



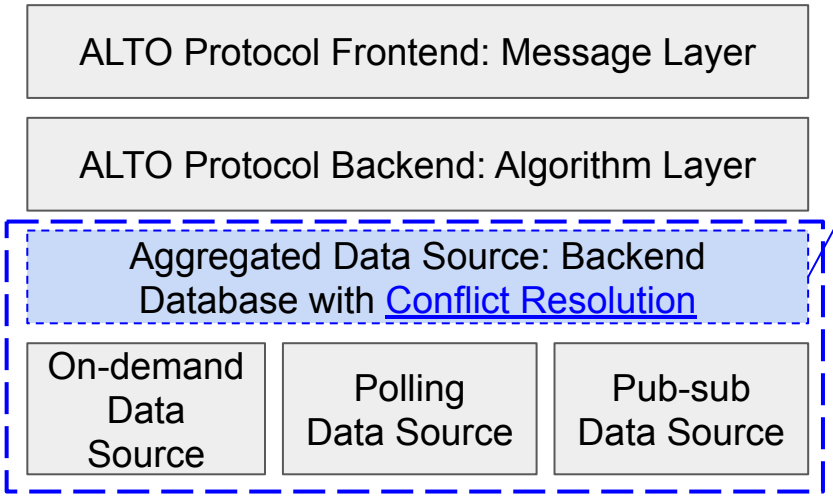
O&M: Common information resource configuration (resource-id, resource-type, capabilities, used algorithm …)

O&M: Implementation-specific configuration parameters (used data sources, PID granularity, cost precision, …)

O&M: Configuration parameters for algorithms to access data sources (southbound protocol, update mechanism, conflict resolution, …)

T3: Connection between Data Sources and Algorithms (cont.)

A main lesson learned from real implementation & deployment:
Different data sources may have conflicts.

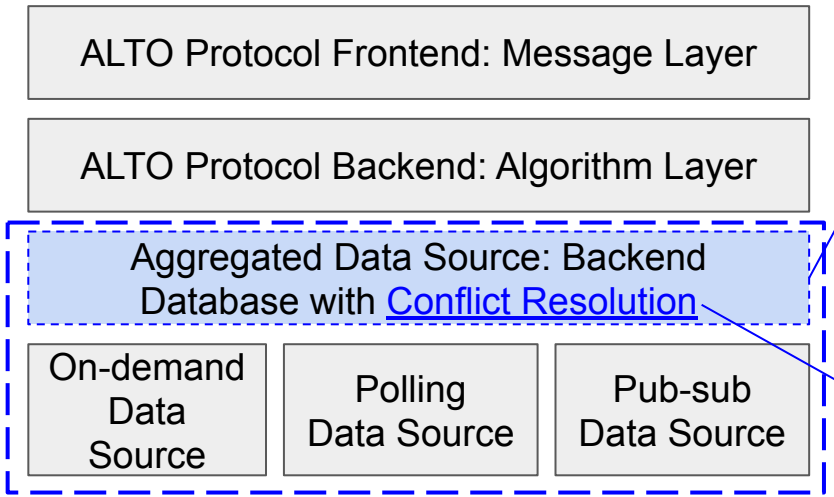


Special data source type: aggregated data source

- An aggregated data source provides a unified data lookup API to other data sources
- Conflict resolution policy can be configured to automatically resolve data source conflicts
- An algorithm can decide whether to use an aggregated data source to resolve conflicts, or handle conflicts by itself.

T3: Connection between Data Sources and Algorithms (cont.)

A main lesson learned from real implementation & deployment:
Different data sources may have conflicts.




```
augment /alto:alto-server/alto:data-source
          /alto:source-params
  +--:(redis-db)
     +--rw redis-params
        +--rw host     inet:host
        +--rw port     inet:port-number
        +--rw db       uint16
        +--rw inputs
        |       -> /alto:alto-server/data-source
        |             /source-id
        +--u conflict-resolver-grouping
```

```
grouping conflict-resolver-grouping
  +-- (conflict-resolver)
     +--:(global-conflict-resolver)
        +-- global-priority* [source-id]
           +-- source-id
           |     -> /alto:alto-server/data-source
           |           /source-id
           +-- priority     uint16
```

Next Step

- Standard track
  - Quickly fix YANG errors and submit a new version.
  - Finish T2.2 and T2.3 soon.
  - YANG doctor review and IESG review?
- Deployment
  - Fully implement O&M in OpenALTO by next IETF.

Backup

Implement ALTO O&M YANG modules in OpenALTO

yang-library-ietf-alto.json > {} ietf-yang-library:modules-state > [ ] module > {} 2

```
{
    "ietf-yang-library:modules-state": {
        "module-set-id": "3de332d13f0da32ea9f00c4b8ae940c6",
        "module": [
            {
                "name": "ietf-alto",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-alto",
                "revision": "",
                "conformance-type": "implement"
            },
            {
                "name": "ietf-alto-stats",
                "namespace": "urn:ietf:params:xml:ns:yang:ietf-alto-stats",
                "revision": "",
                "conformance-type": "implement"
            },
```

alto.conf.template

```
##################################################
# Configure an ALTO server
##################################################
[server]
# Configuration for server setup
default_namespace = default
base_uri = http://openalto.org/
cost_types = {
  "path-vector": {
    "cost-mode": "array",
    "cost-metric": "ane-path"}}

# Configuration for information resources
resources = {
  "directory": {
    "type": "ird",
    "path": "directory",
    "namespace": "default",
    "algorithm": "alto.server.components.backend.IRDService",
    "params": {
      "namespaces": []
    }
  },
```