

# 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-\\_0sY8aI0hGuWbWU8/](https://mailarchive.ietf.org/arch/msg/alto/MvVDgeZnmi-_0sY8aI0hGuWbWU8/)
- 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

1. Introduction	3
2. Requirements Language	3
3. Terminology	3
3.1. Tree Diagrams	3
3.2. Prefixes in Data Node Names	4
4. Design Scope and Requirements	4
4.1. Scope of Data Model for ALTO O&M	4
4.2. Basic Requirements	5
4.3. Additional Requirements for Extensibility	6
4.4. Overview of ALTO O&M Data Model for Reference ALTO Architecture	6
5. Design of ALTO O&M Data Model	6
5.1. Overview of ALTO O&M Data Model	7
5.2. Meta Information of ALTO Server	8
5.3. ALTO Information Resources Configuration Management	10
5.4. Data Sources	12
5.4.1. Yang DataStore Data Source	13
5.4.2. Prometheus Data Source	14
5.5. Model for ALTO Server-to-server Communication	14
6. Design of ALTO O&M Statistics Data Model	14
6.1. Model for ALTO Logging and Fault Management	14
6.2. Model for ALTO-specific Performance Monitoring	14
7. Extension of ALTO O&M Data Model	16
8. ALTO OAM YANG Module	17
8.1. The ietf-alto Module	17
8.2. The ietf-alto-stats Module	34
9. Security Considerations	39
10. IANA Considerations	39
11. References	39
11.1. Normative References	40
11.2. Informative References	41
Appendix A. Example Module for Information Resource Creation Algorithm	42
Acknowledgements	43
Authors' Addresses	43

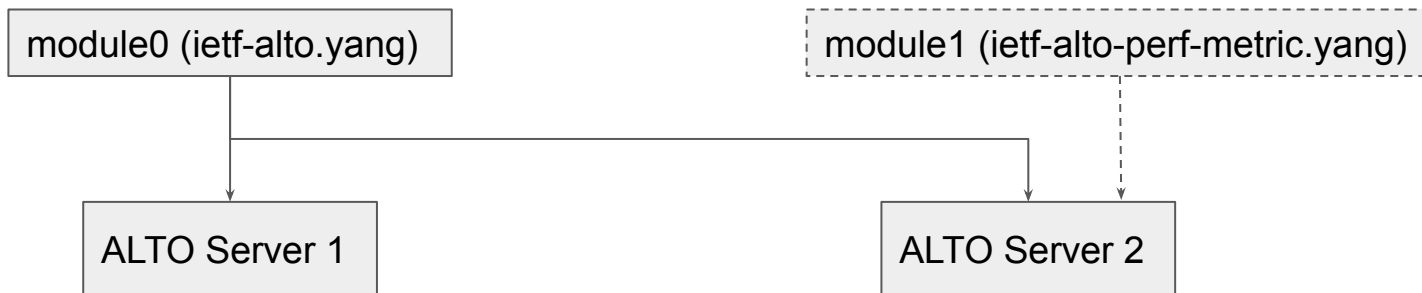
1. Introduction	3
2. Requirements Language	3
3. Terminology	3
3.1. Tree Diagrams	3
3.2. Prefixes in Data Node Names	4
4. Design Scope and Requirements	4
4.1. Scope of Data Model for ALTO O&M	4
4.2. Basic Requirements	5
4.3. Additional Requirements for Extensibility	6
4.4. Overview of ALTO O&M Data Model for Reference ALTO Architecture	6
5. Design of ALTO O&M Data Model	6
5.1. Overview of ALTO O&M Data Model	7
5.2. Data Model for Server-level Operation and Management	8
5.2.1. Data Model for ALTO Server Setup	8
5.2.2. Data Model for Logging Management	9
5.2.3. Data Model for ALTO-related Management	9
5.2.4. Data Model for Security Management	10
5.3. Data Model for ALTO Server Configuration Management	10
5.3.1. Data Source Configuration Management	10
5.3.2. ALTO Information Resources Configuration Management	11
6. Design of ALTO O&M Statistics Data Model	14
6.1. Model for ALTO Server Failure Monitoring	14
6.2. Model for ALTO-specific Performance Monitoring	14
7. ALTO OAM YANG Module	17
7.1. The ietf-alto Module	17
7.2. The ietf-alto-stats Module	34
8. Security Considerations	39
9. IANA Considerations	39
10. References	39
10.1. Normative References	40
10.2. Informative References	41
Appendix A. Example: Extending the ALTO O&M Data Model	42
A.1. Example Module for Extended Data Sources	42
A.2. Example Module for Information Resource Creation Algorithm	42
Acknowledgements	43
Authors' Addresses	43

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.



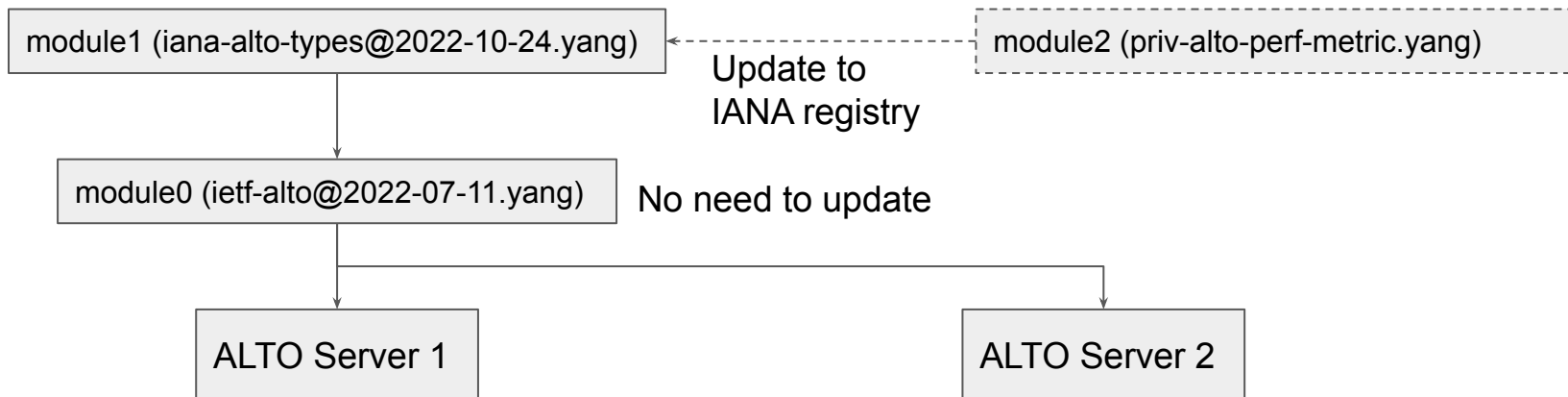
# T1: Data Types in ALTO Related IANA Registries (Cont.)

- 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.

ALTO Information Resource = Algorithm(Data Source 1, Data Source 2, ...)

Configure how to use data

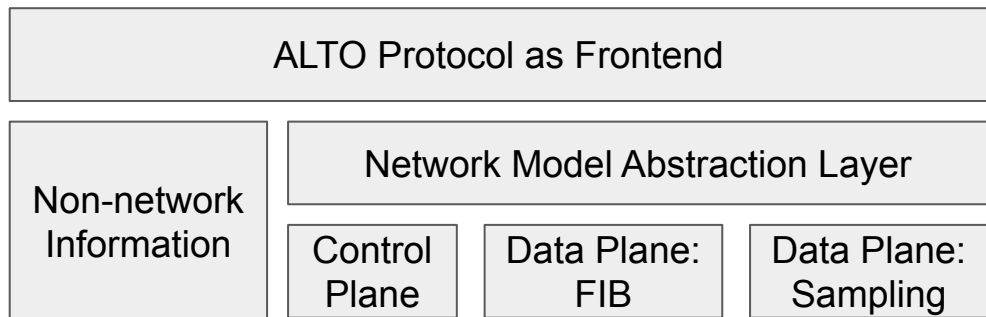
Configure how to get data

O&M: Provide basic, unified model to cover common configuration cases and configuration parameters

# 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

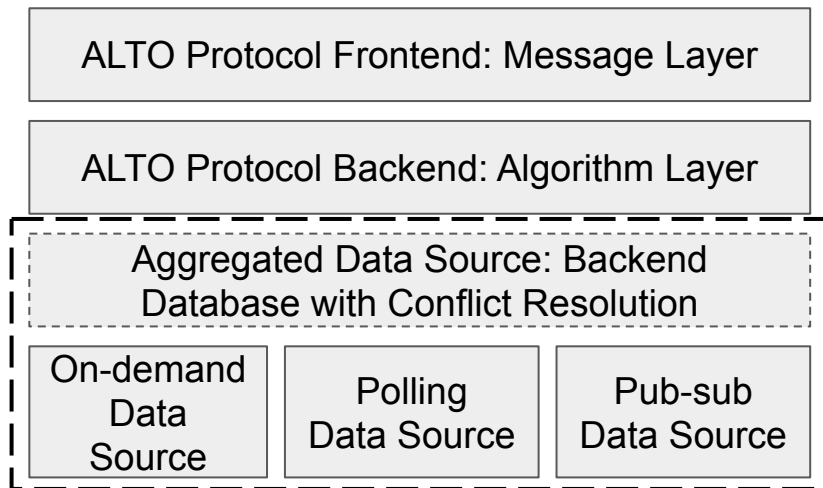


A real deployment in <https://alto.nrp-nautilus.io/directory/default> (IRD):

```
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
```

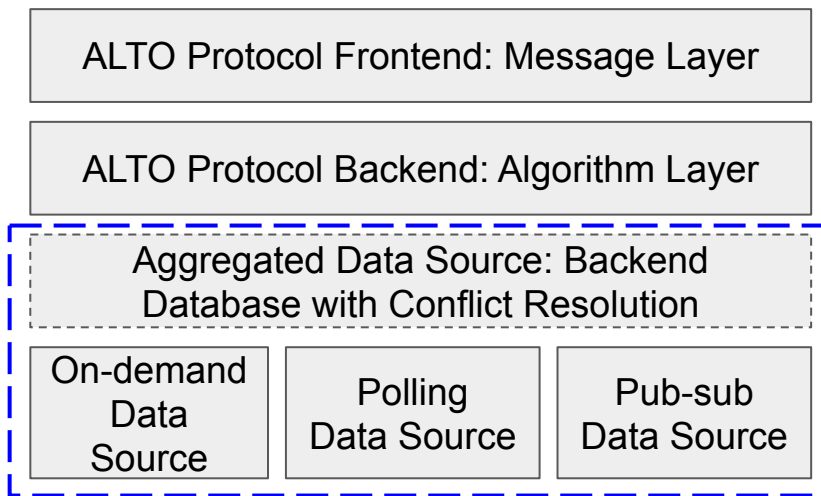
From O&M perspective:

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



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

Mapping from O&M perspective to data model:



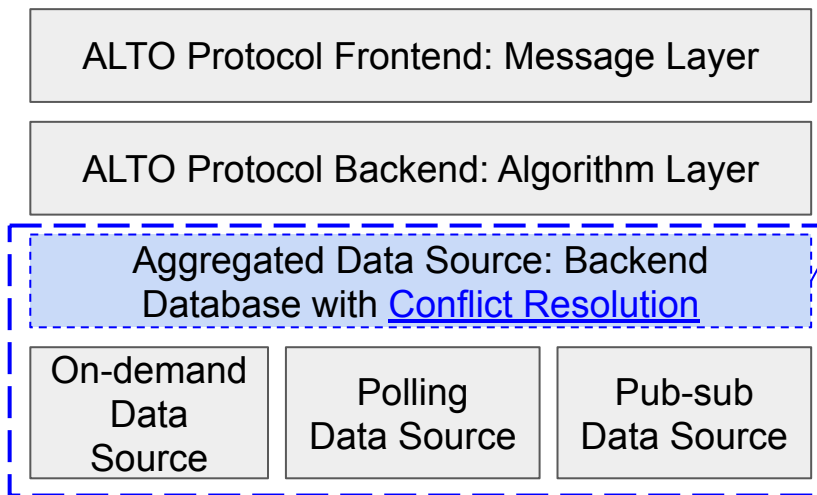
O&M: Common information resource configuration (resource-id, resource-type, capabilities, used algorithm ...)

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.

ALTO Protocol Frontend: Message Layer

ALTO Protocol Backend: Algorithm Layer

Aggregated Data Source: Backend Database with [Conflict Resolution](#)

On-demand  
Data  
Source

Polling  
Data Source

Pub-sub  
Data Source

```
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.IRDSservice",
    "params": {
      "namespaces": []
    }
  }
},
```