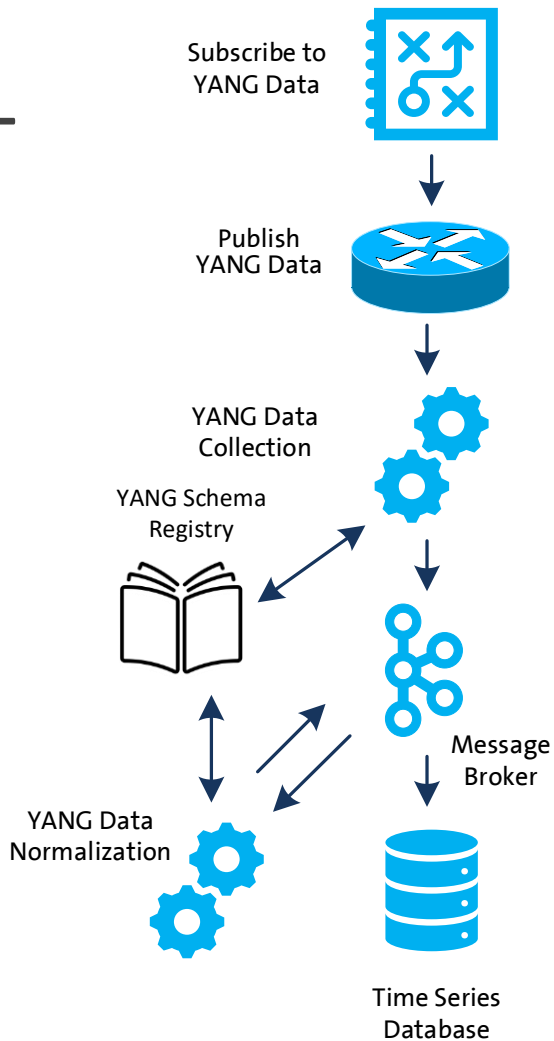


Validate Configured Subscription YANG-Push Publisher Implementations

IETF 123 Hackathon, July 19-20th 2025



Hackathon Plan, Software and Website

Test Plan

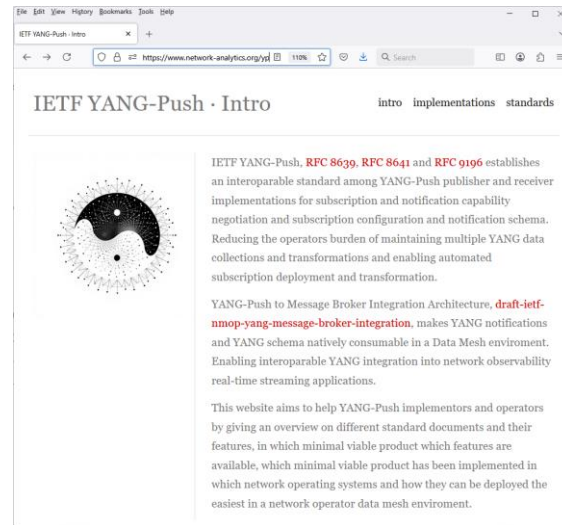
- **Subscription automation**
 - Discover YANG-Push systems and notifications capabilities and configure periodical and on-change subscriptions with netconf.
- **Notification integration**
 - Validate subscription state change and push-update and push-change-update notifications against schema with yanglint
 - Validate [draft-ietf-nmop-message-broker-telemetry-message](#) for [draft-ietf-nmop-yang-message-broker-integration](#) integration

Development Plan

- MVP 1 – Basic Requirements (9)
- MVP 2 – Scale and Secure (3)
- MVP 3 – Optimizations (2)

Software

- YANG-Push Publisher - Cisco IOS XR
- YANG-Push Publisher - 6WIND VSR
- YANG-Push Publisher - Huawei NE (Router) and MA (OLT)
- YANG-Push Receiver – Netgauze
- udp-notif dissector - Wireshark



<https://www.network-analytics.org/yp/how-to-deploy.html>

Hackathon – Repositories


Test Result Repository


- <https://github.com/network-analytics/ietf-network-analytics-document-status/tree/main/123/Hackathon>
 - Packet capture on the wire
 - Netconf RPCs and YANG-Push JSON and CBOR encoded messages

Name	Last commit message	Last commit date
..		
ipf-zbl1243-r-daisy-21_huawei_ma5800t_2025071...	YANG-Push MVP1 Test Results	1 minute ago
ipf-zbl1243-r-daisy-21_huawei_vrp_ne_20250716...	YANG-Push MVP1 Test Results	1 minute ago
ipf-zbl1327-r-daisy-91_cisco_iosxr_20250716_1652...	YANG-Push MVP1 Test Results	1 minute ago
ipf-zbl1843-r-daisy-58_6wind_20250716_165518	YANG-Push MVP1 Test Results	1 minute ago

Test Tool Repository

- https://github.com/network-analytics/yp_test
 - YANG-Push Test Automation Tool
 - Vendor deviations configuration

 ahassany Update readme ✓		
.github/workflows	add back con	
yang-cbor-schema-serde	Use proper st	
yang-cbor-schema-serializer	Use proper st	
yang-json-schema-serde	Use proper st	
yang-json-schema-serializer	Use proper st	
yang-schema-registry-plugin	Use proper st	
.gitignore	Add YANG Ct	
LICENSE	Use apache li	
README.md	Update readme	2 months ago
pom.xml	Use proper semver version 0.0.3	2 months ago

 ybugit Merge pull request #1 from network-analytics/push_to_net... df3b55f · 1 hour ago		
config	push_code	1 hour ago
tests	push_code	1 hour ago
utils	push_code	1 hour ago
vendors	push_code	1 hour ago
LICENSE	Initial commit	yesterday
cli.py	push_code	1 hour ago
device.py	push_code	1 hour ago
main.py	push_code	1 hour ago
readme.md	push_code	1 hour ago
requirements.txt	push_code	1 hour ago
results.py	push_code	1 hour ago
runner.py	push_code	1 hour ago
tee.py	push_code	1 hour ago

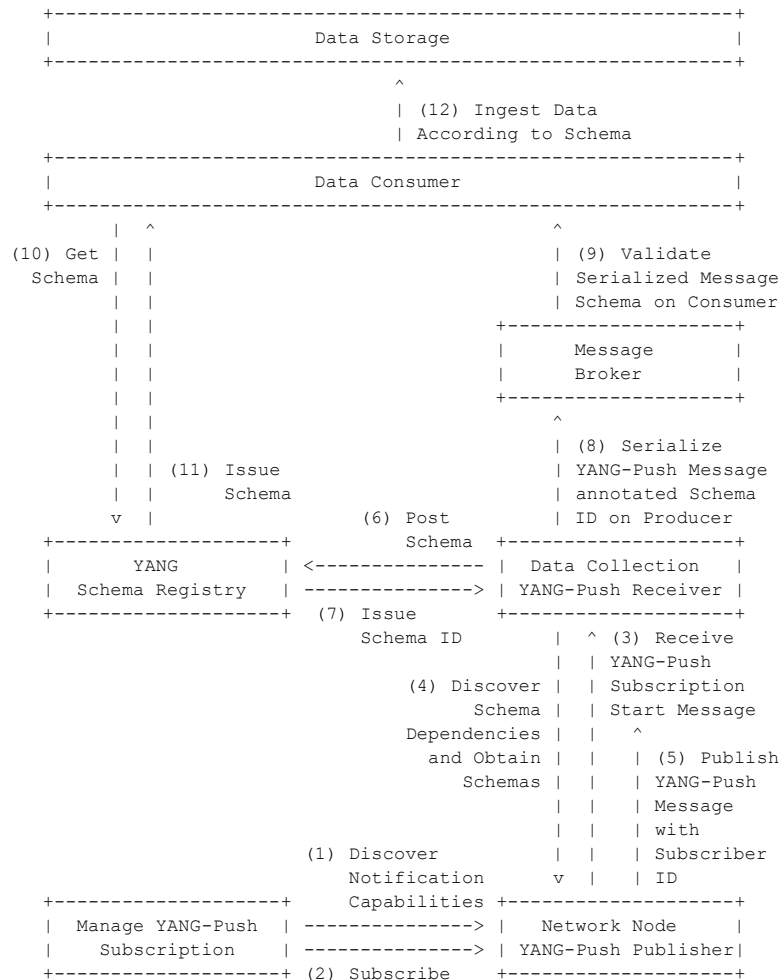
Apache Kafka Integration

- <https://github.com/network-analytics/yang-kafka-integration>
 - YANG Serializer
 - YANG Schema Registry Plugin

An Architecture for YANG-Push to Message Broker Integration

draft-ietf-nmop-yang-message-broker-integration
draft-ietf-nmop-message-broker-telemetry-message

- Subscription to YANG Notifications
[RFC 8639](#)
- Subscription to YANG Notifications for Datastore Updates
[RFC 8641](#)
- UDP-based Transport for Configured Subscriptions
[draft-ietf-netconf-udp-notif](#)
- Subscription to Distributed Notifications
[draft-ietf-netconf-distributed-notif](#)
- Extensible YANG Model for YANG-Push Notifications
[draft-ietf-netconf-notif-envelope](#)
- Support of Versioning in YANG Notifications Subscription
[draft-ietf-netconf-yang-notifications-versioning](#)
- YANG Modules Describing Capabilities for Systems and Datastore Update Notifications
[RFC 9196](#)
- YANG Notification Transport Capabilities
[draft-ietf-netconf-yp-transport-capabilities](#)
- YANG Library
[RFC 8525](#)
- Augmented-by Addition into the IETF-YANG-Library
[draft-ietf-netconf-yang-library-augmentation](#)
- Encoding of Data Modeled with YANG in the CBOR
[RFC 9254](#)



YANG-Push Implementation Status

IETF 123 – MVP 1

	6WIND VSR	Huawei NE	Huawei MA	Cisco IOS XR	Open- Source
RFC 8639 YANG-Push Subscription	✓	✓	✓	✓	
RFC 8641 YANG-Push Notification	✓	P	✓	✓	
draft-ietf-netconf-udp-notif	✓	✓	✓	✓	✓
draft-ietf-netconf-yang-notifications-versioning	✓	✓	✓	✓	
draft-tgraf-netconf-notif-sequencing	✓	✓	✓	✓	
draft-tgraf-netconf-yang-push-observation-time	✓	✓	✓	✓	
RFC 8525 YANG Library	✓	✓	✓	✓	
draft-ietf-netconf-yang-library-augmentation	✓	✓	✓	✓	✓
RFC 9196 System and Notification Capabilities			✓	P	
draft-ietf-netconf-notif-envelope	✓	✓	✓	✓	



[NetGauze](#)
[Pmacct](#)
[udp-notif-c-collector](#)
[yang-library-](#)
[augmentedby](#)

Green marked describes new capability at IETF 123. "P" to partially implemented.

YANG-Push Implementation Status

IETF 123 – MVP 2

	6WIND VSR	Huawei NE	Huawei MA	Cisco IOS XR	Open- Source
draft-ietf-netconf-distributed-notif	✓	✓	✓		
RFC 9254 CBOR Named Identifiers	✓				
RFC 6347/RFC 9147 DTLS					



Green marked describes new capability at IETF 123. "P" to partially implemented

YANG-Push Implementation Status

IETF 123 – MVP 3

	6WIND VSR	Huawei NE	Huawei MA	Cisco IOS XR	Open- Source
RFC 8641 on-change subscriptions	✓	✓	✓	✓	
draft-netana-netconf-yp-transport-capabilities			✓	✓	



Green marked describes new capability at IETF 123. "P" to partially implemented

YANG Library findings - Libyang

Section 3 of RFC 8525 (YANG-Library) describes the relationships between **Datastore <-> Datastore Schema <-> Module Sets <-> Modules** in context of NMDA defined in [RFC 8342](#).

In the hackathon end to end testing, we discovered that [libyang doesn't take datastore and datastore schema into context](#).

```
module: ietf-yang-library
+--ro yang-library
| +--ro module-set* [name] string
| | +--ro name yang:yang-identifier
| | +--ro revision? revision-identifier
| | +--ro namespace inet:uri
| | +--ro location* inet:uri
| | +--ro submodule* [name]
| | | +--ro name yang:yang-identifier
| | | +--ro revision? revision-identifier
| | | +--ro location* inet:uri
| | +--ro feature* yang:yang-identifier
| | +--ro deviation* -> ../../module/name
| +--ro import-only-module* [name revision]
| | +--ro name yang:yang-identifier
| | +--ro revision union
| | +--ro namespace inet:uri
| | +--ro location* inet:uri
| | +--ro submodule* [name]
| | | +--ro name yang:yang-identifier
| | | +--ro revision? revision-identifier
| | | +--ro location* inet:uri
| +--ro schema* [name]
| | +--ro name string
| | +--ro module-set* -> ../../module-set/name
| +--ro datastore* [name]
| | +--ro name ds:datastore-ref
| | +--ro schema -> ../../schema/name
+--ro content-id string
```

libyang / src / context.h

Code Blame 673 lines (624 loc) · 35.1 KB

```
231 * @param[in] options Context options, see @ref contextoptions.
232 * @param[out] new_ctx Pointer to the created libyang context if LY_SUCCESS returned.
233 * @return LY_ERR return value.
234 */
235 LIBYANG_API_DECL LY_ERR ly_ctx_new(const char *search_dir, uint16_t options, struct ly_ctx **new_ctx);
236
237 /**
238 * @brief Create libyang context according to the provided yang-library data in a file.
239 *
240 * This function loads the yang-library data from the given path. If you need to pass the data as
241 * string, use ::ly_ctx_new_ymem(). Both functions extend functionality of ::ly_ctx_new() by loading
242 * modules specified in the ietf-yang-library form into the context being created.
243 * The preferred tree model revision is 2019-01-04. However, only the first module-set is processed and loaded
244 * into the context. If there are no matching nodes from this tree, the legacy tree (originally from model revision 2016-04-09)
245 * is processed. Note, that the modules are loaded the same way as in case of ::ly_ctx_load_module(), so the schema paths in the
246 * yang-library data are ignored and the modules are loaded from the context's search locations. On the other hand, YANG features
247 * of the modules are set as specified in the yang-library data.
248 * To get yang library data from a libyang context, use ::ly_ctx_get_yanglib_data().
249 *
250 * @param[in] search_dir Directory where libyang will search for the imported or included modules and submodules.
251 * If no such directory is available, NULL is accepted.
252 * @param[in] path Path to the file containing yang-library-data in the specified format
253 * @param[in] format Format of the data in the provided file.
254 * @param[in] options Context options, see @ref contextoptions.
255 * @param[in,out] ctx If *ctx is not NULL, the existing libyang context is modified. Otherwise, a pointer to a
```


YANG Library findings – IOS XR/VRP

YANG-Push
subscribes to the
operational
datastore as defined
in [RFC 8641](#).

In the hackathon
end to end testing,
we discovered that
**Cisco IOS XR and
Huawei VRP for
MA5800 misses the
module set,
datastore schema
mapping for
operational
datastore.**

Cisco IOS XR

```
<yang-library xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library">
  <schema>
    <name>All</name>
    <module-set>All</module-set>
  </schema>
  <schema>
    <name>UM-preferred-plus</name>
    <module-set>UM-preferred-plus</module-set>
  </schema>
  <schema>
    <name>UM-preferred</name>
    <module-set>UM-preferred</module-set>
  </schema>
  <schema>
    <name>XR-only</name>
    <module-set>XR-only</module-set>
  </schema>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">idx:running</name>
    <schema>All</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">idx:candidate</name>
    <schema>All</schema>
  </datastore>
</yang-library>
```



Huawei VRP MA

```
<yang-library xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library">
  <schema>
    <name>complete</name>
    <module-set>complete</module-set>
  </schema>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:running</name>
    <schema>complete</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:candidate</name>
    <schema>complete</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:startup</name>
    <schema>complete</schema>
  </datastore>
</yang-library>
```



6WIND VSR

```
<yang-library xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library">
  <schema>
    <name>complete</name>
    <module-set>complete</module-set>
  </schema>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:running</name>
    <schema>complete</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:candidate</name>
    <schema>complete</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:startup</name>
    <schema>complete</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:operational</name>
    <schema>complete</schema>
  </datastore>
</yang-library>
```



Huawei VRP NE

```
<yang-library xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-library">
  <schema>
    <name>config-schema</name>
    <module-set>config-module</module-set>
  </schema>
  <schema>
    <name>state-schema</name>
    <module-set>config-module</module-set>
    <module-set>state-module</module-set>
  </schema>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:startup</name>
    <schema>config-schema</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:running</name>
    <schema>config-schema</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:candidate</name>
    <schema>config-schema</schema>
  </datastore>
  <datastore>
    <name xmlns:ds="urn:ietf:params:xml:ns:yang:ietf-datastores">ds:operational</name>
    <schema>state-schema</schema>
  </datastore>
</yang-library>
```



YANG Data Structure Extensions - Yanglint

In IETF 122 hackathon we discovered that YANG Data Structure Extensions [are supported in libyang but not in yanglint](#).

YANG Data Structure Extensions support in yanglint has recently been added and validated in this hackathon. However, [there are still issues](#).



mxyns 19 hours ago

I've also tried with

```
{
  "example-module:address": [
    {
      "city": "Bedrock",
      "first": "Fred",
      "last": "Flintstone",
      "street": "301 Cobblestone Way"
    },
    {
      "city": "Bedrock",
      "first": "Charlie",
      "last": "Root",
      "street": "4711 Cobblestone Way"
    }
  ]
}
```

and I get the following error

```
YANGLINT[E]: Yanglint does not support more than one top-level node in extension data.
YANGLINT[E]: Failed to parse input data file "/home/max/yangshi/example.json".
```



Register new YANG schema - Payload

The same rest API as other formats such as AVRO, JSON, and ProtoBuf is used to register new YANG schemas.

Hackathon 124 proposal: Lets Support optional YANG features ([Section 5.6.2 in RFC 7950](#)) in YANG schema registry.

```
curl -X POST \  
  -H "Content-Type: application/vnd.schemaregistry.v1+json" \  
  -d @my-module-request.json \  
  http://localhost:8081/subjects/my-module/versions
```

```
{  
  "schemaType": "YANG",  
  "references": [  
    {  
      "name": "other-module-name",  
      "subject": "registered subject name",  
      "version": "registered version",  
    }  
  ],  
  "metadata": {  
    "tags": {  
      "features": ["arbitrary-names", "if-mib"]  
    }  
  },  
  "schema": "... yang schema text"  
}
```

my-module-request.json

Retrieve new YANG schema

- Retrieve all registered schemas
curl <http://localhost:8081/subjects/>
- Retrieve all registered version of a given subject
curl <http://localhost:8081/subjects/my-module>
- Retrieve a specific version of a schema registry
curl <http://localhost:8081/subjects/my-module/versions/1>
- Retrieve a schema by ID
<http://localhost:8081/schemas/ids/1>

```
{
  "schemaType": "YANG",
  "subject": "my-module",
  "version": 1,
  "id": 1,
  "references": [
    {
      "name": "other-module-name",
      "subject": "registered subject name",
      "version": "registered version",
    }
  ],
  "metadata": {
    "tags": {
      "features": ["arbitrary-names", "if-mib"]
    }
  },
  "schema": "... yang schema text"
}
```

Result of getting schema from schema registry

Apache Kafka wire format

- Data is encoded in native YANG format (JSON, CBOR).
- Schema ID is included in the header.
- Content type is encoded in the header using the standard allocated in IANA ([RFC 8040](#) and [RFC 9254](#))

```
{  
  "headers": [  
    "schema-id": 1, // schema id registered  
    "content-type": "application/yang-data+json"  
  ]  
  "payload": ..json encoded YANG,  
}
```

Apache Kafka message value and headers

Next Steps...

- [Yanglint](#), fix YANG extension's structure support
- [YANG schema registry](#) supporting YANG features
- [NetGauze](#) supporting
 - netconf <get> to obtain YANG library content,
 - YANG validation for YANG-Push notifications,
 - YANG schema registration.
- [Pmacct](#) supporting Network Telemetry Message

Thanks to...

- Rob Wilton – Cisco
- Dan Voyer – Cisco
- Nick Corran – Cisco (remote)
- Emma Rankin – Cisco (remote)
- Mathew Green – Cisco (remote)
- Samuel Gauthier – 6WIND (remote)
- Jérémie Leska – 6WIND (remote)
- Liu Bin – Huawei (remote)
- Benoit Claise – Huawei
- Zhuoyao Lin – Huawei (remote)
- Jiale Li – Huawei (remote)
- Jian Ping – Huawei (remote)
- Xiao Chen – Huawei (remote)
- Paolo Lucente – Pmacct
- Holger Keller – DT
- Nils Warnke – DT
- Alex Huang-Feng – INSA Lyon
- Maxence Younsi – INSA Lyon
- Vivekananda Boudia – INSA Lyon
- Pierre Francois – INSA Lyon
- Boris Hassanov – MWS
- Yannick Buchs – Swisscom (remote)
- Ahmed Elhassany – Swisscom
- Thomas Graf – Swisscom



Want to know more than join us at...

- NMOP on **Monday 09:30 – 11:30** at Castilla for Message Broker Telemetry Message
- NMOP on **Wednesday 16:00 – 17:00** at Castilla for more on YANG-Push validation
- Side Meeting on **Friday 09:30 – 10:30** at El Escorial for the How to Use YANG-Push MVP 1 with Message Broker Integration Demo.