Validating anydata in YANG Library context

draft-netana-nmop-yang-anydata-validation

ahmed.elhassany@swisscom.com
Thomas.graf@swisscom.com

21. July 2025

Context

- RFC 7950: The YANG 1.1 Data Modeling Language

  The "anydata" statement is used to represent an unknown set of nodes that can be modeled with YANG, except anyxml, but for which the data model is not known at module design time. It is possible, though not required, for the data model for anydata content to become known through protocol signaling or other means that are outside the scope of this document.

Where anydata is currently used?

incomplete list

- RFC 8342: ietf-netconf-nmda
- RFC 9144: ietf-nmda-compare
- RFC 8040: ietf-restconf
- RFC 8639: ietf-subscribed-notifications
- RFC 9195: ietf-yang-instance-data
- RFC 8072: ietf-yang-patch
- RFC 8072: ietf-yang-push
- RFC 8532: ietf-connectionless-oam (uses yang mount)
- RFC 8791: any YANG data structure is encoded the same way as anydata node.

Problem statement

- How can we validate the schema subtree of an anydata node?

```
notifications:
    +---n push-update
    |   +--ro id?                      sn:subscription-id
    |   +--ro datastore-contents?   <anydata>
```

Schema definition of push-update notification

```
{
 "ietf-yang-push:push-update": {
  "id": 89,
  "datastore-contents": {
   "ietf-interfaces:interfaces": {
   "interface": [
    {
     "name": "eth0",
     "oper-status": "down"
    }]
   }}
  }
}
```

Example Message

YANG Library look up

- The namespace of the encoded data nodes under anydata can be looked up in a YANG Library context.

```
{
"ietf-yang-library:yang-library": {
 "module-set": [
  {
   "name": "complete",
   "module": [
    {
     "name": "yang",
     "revision": "2022-06-16",
     "namespace": "urn:ietf:params:xml:ns:yang:1"
    },
    {
     "name": "ietf-interfaces",
     "revision": "2018-02-20",
     "namespace": "urn:ietf:params:xml:ns:yang:ietf-interfaces",
     "location": ["file://ietf-interfaces@2018-02-20.yang"],
     "feature": [
       "arbitrary-names",
       "pre-provisioning",
       "if-mib"
     ]
    },
  …
```

Changes since IETF 119

1. Clarify the language of validation option and use the terms defined in RFC 7950:
    1. Complete validation: validates the contents of the anydata subtree, which MUST obey all validation rules defined in the corresponding schema in the YANG Library.
    2. Candidate validation: validation without applying not apply the constraint checks.
2. Test the libyang implementation with YANG Push (RFC 8072) and draft-ietf-nmop-message-broker-telemetry-message.

Implementation

- Current libyang implementation disables strict parsing while in anydata subtree. Implementing this draft would require to change this behavior with an optional flag and use strict validation always.

```
diff --git a/src/parser_xml.c b/src/parser_xml.c
index 5d97c8e49..6938d3712 100644
--- a/src/parser_xml.c
+++ b/src/parser_xml.c
@@ -931,7 +931,7 @@ lydxml_subtree_any(struct lyd_xml_ctx *lydctx, const struct lysc_node *snode, co
        LY_CHECK_ERR_GOTO(r, rc = r, cleanup);

        /* update options so that generic data can be parsed */
-       lydctx->parse_opts &= ~LYD_PARSE_STRICT;
+       //lydctx->parse_opts &= ~LYD_PARSE_STRICT;
        lydctx->parse_opts |= LYD_PARSE_OPAQ | (ext ? LYD_PARSE_ONLY : 0);
        lydctx->int_opts |= LYD_INTOPT_ANY | LYD_INTOPT_WITH_SIBLINGS;
```

Question and next steps

- We request adoption from the NETMOD
- Extend the libyang implementation to support complete validation.
- Push changes to libyang and make them accessible via yanglint.