YANG Schema Comparison

draft-ietf-netmod-yang-schema-comparison-02

IETF 116 NETMOD WG 2023-03-31

Joe Clarke <jclarke@cisco.com> (Presenter, Contributor)
Rob Wilton <rwilton@cisco.com> (Author)
Per Andersson <perander@cisco.com> (Editor)

YANG Schema Comparison

- Tool requirements for comparing YANG Schema
- Help developers to categorize changes as BC, NBC, Editorial etc.
- Help users to identify if they are affected by NBC changes

Current Key Issues

- On-wire vs Schema analysis
- Compatibility of Error Messages
  - Error messages, error-tags, and other error statements
- Comparison Scope
  - Compare on module or full schema (YANG artifact, arbitrary YANG blob)

On-wire vs Schema Analysis

- One or two algorithms?
  - Consensus reach was two different
- On-wire: Focus on client compatibility
- Schema: Any change according to ietf-netmod-yang-module-versioning
- Non semantic changes: indicate editorial change occurred
- Reordering not allowed, as per RFC 7950

On-wire Algorithm

- Comparison is made by traversing the flattened tree, i.e. all paths, of the YANG definitions
  - Instance identifier is used (e.g. choice names omitted)
- For each path, check if any property has an NBC change
  - NBC with regards to the client’s usage of the model
  - Properties such as base type and range
  - Added or removed paths
- “choice”, “grouping”, “typedef” rename has no effect

Schema Algorithm

- Comparison is made by traversing the old and new YANG definition hierarchies
  - Schema node identifier is used (e.g. choice names included)
- Any change is reported
  - E.g. typedef name changes
- The full NBC rules of draft-ietf-netmod-yang-module-versioning apply

Example Algorithm Application

- Schema, e.g. all changes are
  - leaf counter { type uint8; }
  - leaf counter { type counter_t; }
  - Would be non-backwards-compatible
- On-wire, e.g. comparing messages
  - <counter>42</counter>
  - Would be backwards-compatible

On-wire vs Schema analysis, cont’d

- Several open questions still exist
- Are they deemed to be NBC?
- How and/or where should they be reported
  - at definition, at usage, both?

On-wire vs Schema analysis, cont’d

- Open questions (NBC? How to report?)
  - Groupings / uses
  - Typedefs, namespaces, choice names, prefixes, module metadata
  - Typedef renaming (on-wire, same base type etc)
  - Should all editorial (text) diffs be reported?

On-wire vs Schema analysis, cont’d

- Open questions (NBC? How to report?)
  - Editorial changes that change semantics, e.g. a description (for behavior) of a leaf.
    - Tune verbosity: whitespace, spelling, editorial, potentially-NBC
  - XPath, “must”, “when”
    - don’t normalize expressions, text diff comparison is made
    - If change is detected, mark as NBC or potentially-NBC
  - Presence statements

Compatibility of Error Messages

- Error tags and messages might be relied on verbatim by users
  - error-tag: standardized in RFC 6241
  - error-app-tag: standardized in RFC 6241 (and model)
  - error-message: arbitrary
- Failed “must” statement, error-message, assumed NBC
- Default behavior is that changes to error tags, messages etc are NBC.

Comparison Scope

- What scope is the comparison made?
  - Packages vs directories vs libraries vs artifact vs arbitrary YANG blob
  - Features
  - For package specific comparison, compare the bundled metadata as well or only the modules?
  - Import only or implemented module?
- Filter out comparison for a specific subtree, path etc
  - Use case: on-wire for e.g. YANG subscriptions, did the model change for my subscription?

Open Issues

- Override/per-node tags
- Separate rules for config vs state
- Tool/report verbosity
- Sub-modules
- Publish algorithm in pseudo code or text
- Report categories and/or filters
  - bc, nbc, potentially-nbc, editorial
- Only for YANG 1.1?
- Renamed-from

Thank you!

Questions?