NETMOD YANG Versioning Solution Update

NETMOD WG
July 2023

Presenting on behalf of the weekly versioning call attendees:
Joe Clarke

IETF 117
Agenda

This presentation:
  Solution Overview
  Key Issues from WG LC of Module Versioning and YANG Semver

See backup slides for solution overview & summary of Module Versioning and YANG Semver drafts.
Recap - YANG Versioning Solution Overview

Complete solution consists of five drafts:

1. Updated YANG Module Revision Handling:
   Notify nbc changes between module revisions, allows branched revision history, revision-labels

2. Module semantic version number scheme:
   YANG semver for module revision-labels and package versioning

3. YANG schema comparison tooling:
   Tooling to algorithmically compare module or schema revisions

4. Versioned YANG packages:
   Versioning at the schema level rather than individual modules

5. Protocol operations for package version selection:
   Devices can support multiple schema versions, clients can select for session

Latest working drafts can all be found here: https://github.com/netmod-wg/yang-ver-dt
Recap - YANG Versioning Solution Overview

Links for published drafts:

https://datatracker.ietf.org/doc/draft-ietf-netmod-yang-schema-comparison
General update on weekly Versioning calls

• Authors + interested parties continue to meet on a weekly call to progress this work
  • Meetings are open to all
  • Regular participation from people across 5+ different companies (mostly equip. vendors, participation from more operators/users would be beneficial)
  • Key issues are brought back to WG mailing list
  • Weekly meeting is currently on Tues @2pm UK time / 9am Eastern. Thanks to the authors and contributors for their regular attendance

• Issues tracked in github (https://github.com/netmod-wg/yang-ver-dt)
General update on weekly Versioning calls

2nd WG LC launched for Module Versioning + YANG Semver

Focus of weekly meetings since IETF116 has been primarily on feedback from the 2nd WG LC:
  • Several key/fundamental issues being discussed/debated (upcoming slides...)

Some limited time spent on Schema Comparison draft but still a lot of work to do.

Packages draft is the next focus after Schema Comparison.

Version Selection hasn’t been looked at in quite a long time.
Key Issues from
WG LC of
Module Versioning and
YANG Semver
Key Issue 1

Allow NBC changes in YANG?
Option 1: Update RFC7950 to Allow NBC Changes

- Module Versioning modifies 7950 to allow NBC changes
- guidance that NBC changes SHOULD NOT be done (impact to user base)
- rev:non-backwards-compatible is a YANG extension
  - introduction in published YANG does not impact current tooling (ignored until recognized)

**PROS:**
- address fundamental requirement of this versioning work (requirements doc)
- allows gradual adoption in the industry. YANG authors can immediately start publishing with the new extensions.
- move faster to produce modules in the IETF (accept some errors/iteration)
- address the liaison from external standards bodies in a reasonable timeframe
- authors believe work is ready
- broad vendor support
- rough alignment with OpenConfig (use YANG 1.0 + OC Semver)

**CONS:**
- perception that we're "cheating" by not bumping our own spec's version
- Not fundamentally mandatory for clients or servers using YANG (mandatory for YANG claiming conformance to Module Versioning).
Option 2: RFC7950-bis: Publish a new version of the YANG language to allow NBC changes

- NBC changes only allowed in a new (future) version of YANG
- TBD: YANG 1.2 vs 2.0 (note YANG 1.1 isn't BC with YANG 1.0)
- Content = Module Versioning + YANG Semver + very limited YANG NEXT items
- rev:non-backwards-compatible tag is a language keyword
  - consequence: any use of it breaks all YANG 1.0/1.1 tooling that hasn't been updated
- TBD how to handle small NBC changes in IETF in the short term (i.e. non conformance to 7950)?
  - RFC6991 bis - change the use/meaning of ip-address (or change datetime)
    - YANG date-and-time (because of SEDATE date string changes)

**PROS:**
- address fundamental requirement of this versioning work (requirements doc)
- clear delineation of changes in the YANG language
- consistent with philosophy that version number changes for significant changes in a spec (avoids concern that YANG is changing without bumping the version of YANG)
- can do this with mandatory YANG keywords which helps increase conformance to the new rules

**CONS:**
- difficult to roll out in the industry. Tools need upgrading before they won’t error on a YANG 1.2 module.
- Authors can't publish YANG 1.2 until their users have upgraded their tools. Everyone has to move at once.
- likely large delay in producing the work (unclear what would go into YANG 1.2, may not reach consensus easily on N items)
- delay in follow up work (Packages, Schema Comparison, Version Selection)
- continue dominating WG effort for longer (opportunity cost)
Option 3: Strict Adherence to Current RFC7950 Rules

- IESG will be unable to approve any RFCs that make any changes to IETF YANG modules that don’t strictly conform to those rules
  - RFC6991 bis would not be allowed to change the use/meaning of ip-address (or change datetime)
  - YANG date-and-time couldn't change (related to SEDATE date string changes)

**PROS:**
- clear rules for entire industry including IETF

**CONS:**
- doesn't address agreed/adopted requirements of YANG versioning work
- incorrect assumption in tool chains, etc that NBC changes don't happen. Silent failures.
Key Issue 2

single v/s multiple revision label schemes
Recap of *revision-label-scheme*

- Extension defined in YANG module versioning document.
- Takes a mandatory parameter defining the scheme used, it is an identity derived from *revision-label-scheme-base*
- Extension MUST be used if there is a revision label statement in the (sub)module
- The YANG Semver document defines the scheme *yang-semver*
  (note – the current YANG revision date is not considered a revision label / label scheme)

Example:
```
rev:revision-label-scheme "yangver:yang-semver";
```
Pros of \textit{revision-label-scheme}

• YANG Semver deemed too restrictive by some

• This provides flexibility to e.g. have vendor specific schemes which allow for infinite branching where the versions have no semantic meaning

• Consistent framework for adding other schemes
Cons of *revision-label-scheme*

- Flexibility comes with cost of added complexity, e.g. what if a module changes from scheme A to scheme B
- YANG Semver is sufficient for IETF and many vendors
- If some entity wants their own scheme they could just do it using their own separate extension (outside of any “framework”)
Impact of removing *revision-label-scheme*

- We would rename *revision-label* e.g. to *yangsemver-label*
- If a vendor wants a new versioning scheme, a proprietary extension would need to be added by that vendor (including augmentations of yang library, packages, etc)
- The current IETF documents would be simpler
- Cost/effort to make the changes to the documents
Key Issue 3

Why do we need YANG Semver (vs. SemVer 2.0.0)?
SemVer 2.0.0

• Linear (no branching)
• Simpler in construction
  • Major
  • Minor
  • Patch
• 1.0.0, 1.0.1, 1.1.0, 2.0.0, ...
  • If a new feature is needed in 1.0.1, a 1.2.0 would need to be minted that incorporates the features of 1.1.0
• Widely liked by the industry, but only works well when updating at the head (fine for open source, not acceptable for operators)
YANG Semver

• Support for *limited* branching (maintenance of released code)
• Supports SemVer 2.0.0 rules
• MAJOR.MINOR.PATCH_MODIFIER
  • _compatible
  • _non_compatible

• Example:
  
  1.0.0
  1.0.1 -- 1.0.2_non-compatible
  1.1.0
  2.0.0

• A feature (or an NBC change can be backported)
Why YANG Semver

• Given that module versioning allows branching, the labeling scheme must also support branching

• YANG Semver is a compromise between power and simplicity
  • Encourage “mostly” single track development with modifiers the exception
  • Retains support for some updates to older versions

• Sufficient for SDOs and vendors

• Industry is familiar with Semver – tried to stay close to it
BACKUP SLIDES
Module Revision Handling + YANG Semver Drafts
Summary of main requirements & recommendations

draft-ietf-netmod-yang-module-versioning - Updated YANG Module Revision Handling
draft-ietf-netmod-yang-semver - YANG Semantic Versioning

A) Mark a revision as non-backwards-compatible

```yang
module example-module {
  namespace "urn:example:module";
  prefix "prefix-name";

  import ietf-yang-revisions { prefix "rev"; }
  import ietf-yang-semver { prefix "ysver"; }

  rev:revision-label-scheme "ysver:yang-semver";

  revision 2019-06-01 {
    rev:revision-label 3.1.0;
    description "Add new functionality.";
  }

  revision 2019-03-01 {
    rev:revision-label 3.0.0;
    rev:non-backwards-compatible;
    description "Add new functionality. Remove some deprecated nodes.";
  }

  revision 2019-02-01 {
    rev:revision-label 2.0.0;
    rev:non-backwards-compatible;
    description "Apply bugfix to pattern statement";
  }
}
```
B) Recommending a minimum revision for module imports

Specify a *minimum* revision for import, or any descendant of the minimum

Documentation of a recommendation. Not a strict conformance rule.

```
import example-module {
  rev:recommended-min 3.0.0
}
```
Module Revision Handling + YANG Semver Drafts
Summary of main requirements & recommendations

C) YANG Semver revision-label

X.Y.Z = Major.Minor.Editorial

Major -> change for NBC changes
Minor -> additional leaves, etc
Editorial -> fix a description

Example YANG module with branched revision history.

<table>
<thead>
<tr>
<th>Module revision date</th>
<th>Revision label</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-01-01</td>
<td>&lt;= 1.0.0</td>
</tr>
<tr>
<td>2019-02-01</td>
<td>&lt;= 2.0.0</td>
</tr>
<tr>
<td>2019-03-01</td>
<td>&lt;= 3.0.0</td>
</tr>
<tr>
<td>2019-04-01</td>
<td>&lt;= 2.1.0</td>
</tr>
<tr>
<td>2019-05-01</td>
<td>&lt;= 2.2.0</td>
</tr>
<tr>
<td>2019-06-01</td>
<td>&lt;= 3.1.0</td>
</tr>
</tbody>
</table>

Optional compatibility extension/suffix:
rev:revision-label 3.0.2_non_compatible
Module Revision Handling + YANG Semver Drafts
Summary of main requirements & recommendations

D) Filename recommendation

my-module#3.2.0.yang