YANG Packages

draft-ietf-netmod-yang-packages

NETMOD WG
Nov 2020

Presentation Authors: Bo Wu & Jason Sterne
Presenting: Bo Wu
YANG Package Current Status –

• Weekly call of the WG versioning meeting since IETF 108
  • Quite some time was spent on the package discussion

• Changes since -00
  • New deviation rule for 5.1 Package definition rules: describe deviation statement in YANG packages, including the relationship with RFC7950 deviation definition and restrictions on the SDO use of deviation
  • New Section 5.8. YANG Package Usage Guidance: guideline on the use of YANG Packages, and also deviations and features in YANG packages
  • Many editorial changes based on the comments from weekly call and WG mailing list, for better reading and consistent with other Versioning drafts

• Ongoing discussion
  • Open issues, 16 total open issues now
  • https://github.com/netmod-wg/yang-ver-dt/issues?q=is%3Aopen+is%3Aissue+label%3Apackages
YANG Package Usage Guidance

• General guidelines
  1. Organizations is recommended to publish YANG package definition for better interoperability
  2. Deviations should be reserved for implementations rather than specifications, and publishing (standardizing) a package that is based on another standard package with some deviations is very dangerous.
  3. Published standard YANG modules SHOULD make use of appropriate feature statements to provide flexibility
Open Issues
Packages can contain modules with mount points.

Should there be some additional information in the package definition about mount points, where they are, what can be mounted?

For “what can be mounted”
- a list of packages?
- a minimum set of modules/packages?
- a set of constraints on what is allowed at the mount points?

In the weekly versioning calls we currently feel that this is “in scope” for Packages. It is useful for a server implementation to be able to describe (e.g. offline) the full schema it supports (which includes schema that is mounted).
Require module revision date


Should we require that all modules in a package have at least a revision date? MUST vs SHOULD.

YANG allows modules without a revision date.

One option: if a module revision is not specified then it only identifies the YANG module that has no revision identifier.

- Couldn’t there be multiple versions without a revision identifier?
Allow circular YANG package inclusion?

https://github.com/netmod-wg/yang-ver-dt/issues/64

Allow circular package inclusion and clients/servers/tools can truncate inclusion chains when a repetition is hit?

• Note: this would not change the prohibition against circular includes in *modules*

Blocking circular inclusion may prevent different groups from independently defining their packages.

• E.g. could a routing package depend on an OAM package?
• Could an OAM package also depend on a routing package?

This will be further discussed in weekly calls before coming back to the WG.
Where should YANG packages guidelines text go? (GitHub Issue #71)

• Should the individual drafts update YANG author guidelines or should we try and do a bis update of YANG author guidelines?

• YANG package guidelines don't obviously fit into the scope of the YANG author guidelines for modules. Possibly it would be better to have a separate section as part of the YANG packages document covering implementation guidelines and recommendations.

• **Proposal: define a new section in Packages draft**

Backup
**YANG Package Model Structure**

**Common package definition for both the YANG instance data file and the device**

```
grouping yang-pkg-instance
  +-- name               pkg-name
  +-- version            pkg-version
  +-- timestamp?         yang:date-and-time
  +-- organization?      string
  +-- contact?           string
  +-- description?       string
  +-- reference?         string
  +-- complete?          boolean
  +-- local?             boolean
  +-- previous-version?  pkg-version
  +-- nbc-changes?       boolean
  +-- tag*               tags:tag
  +-- mandatory-feature* scoped-feature
  +-- included-package*  [name version]
    |   +-- name         pkg-name
    |   +-- version      pkg-version
    |   +-- replaces-version* pkg-version
    |   +-- nbc-modified? boolean
    |   +-- location*    inet:uri
    |   +-- checksum?    pkg-types:sha-256-hash
  +-- module* [name]
    |   +-- name         yang:yang-identifier
    |   +-- revision      rev:revision-date-or-label
    |   +-- replaces-revision* rev:revision-date-or-label
    |   +-- namespace?    inet:uri
    |   +-- location*     inet:uri
    |   +-- checksum?     pkg-types:sha-256-hash
    +-- submodule* [name]
      |   +-- name?        yang:yang-identifier
      |   +-- revision?    yang:revision-identifier
      |   +-- location?    inet:uri
      |   +-- checksum?    pkg-types:sha-256-hash
  +-- import-only-module* [name revision]
    |   +-- name?         yang:yang-identifier
    |   +-- revision?     rev:revision-date-or-label
    |   +-- replaces-revision* rev:revision-date-or-label
    |   +-- namespace?    inet:uri
    |   +-- location?     inet:uri
    |   +-- checksum?     pkg-types:sha-256-hash
  +-- submodule* [name]
    |   +-- name?        yang:yang-identifier
    |   +-- revision?    yang:revision-identifier
    |   +-- location?    inet:uri
    |   +-- checksum?    pkg-types:sha-256-hash
```

**Package list on Server: all versions of all packages supported**

```
module: ietf-yang-packages
  +--ro packages
    +--ro package* [name version]
      // Uses the yang-package-instance grouping defined in
      // ietf-yang-package-types.yang, with location and checksum:
      +--ro name                pkg-name
      +--ro version             pkg-version
      ... remainder of yang-package-instance grouping ...
      +--ro location*           inet:uri
      +--ro checksum?           pkg-types:sha-256-hash
```

**YANG Library augmentation:**

Clients generally uses the packages to associate with the datastore schema instead of the module set

```
module: ietf-y1-packages
  augment /yanglib:yang-library/yanglib:schema:
    +--ro package* [name version]
      +--ro name     -> /pkgs:packages/package/name
      +--ro version  leafref
      +--ro checksum? leafref
```