NETMOD Versioning Design Team Update

NETMOD WG

March 25, 2019

Netmod YANG Version Design Team

Presenting: Joe Clarke, Rob Wilton, Balazs Lengyel, Reshad Rahman

Agenda

- **Requirements Draft update – Joe Clarke (5 mins)**
  draft-verdt-netmod-yang-versioning-reqs-02
- **Design Team update & Solution overview – Rob Wilton (10 mins)**
- **Semantic versioning for modules – Balazs Lengyel (20 mins)**
  draft-verdt-netmod-yang-semver-00
- **Semantic versioning for YANG schema – Rob Wilton (10 mins)**
  draft-rwilton-netmod-yang-packages-01
- **Schema version selection – Reshad Rahman (10 mins)**
  draft-wilton-netmod-yang-ver-selection-00

draft-verdt-netmod-yang-versioning-reqs-02

Netmod WG

March 25, 2019

Netmod YANG Version Design Team

Joe Clarke (presenting)

Changes Since -01

- Clarify requirement 1.4 to remove wording around “software releases”
- Clarify requirement 3.2 so that it does not suggest a solution but merely describes the problem of needing to support older clients expecting older NBC-changed modules
- Remove requirement 4.4
- Acknowledge the inspiration from OpenConfig

Requirement 1.4

OLD:

“The solution MUST allow for modules to be versioned by software release. In particular, backwards-compatible enhancements and bug fixes MUST be allowed in any non-latest release.”

NEW:

“The solution MUST allow for backwards-compatible enhancements and bug fixes, as well as non-backwards-compatible bug fixes in non-latest-release modules.”

Requirement 3.2

OLD:

“The solution MUST provide a mechanism to allow servers to simultaneously support clients using different revisions of modules. A client's choice of particular revision of one or more modules may restrict the particular revision of other modules that may be used in the same request or session.”

NEW:

“The solution MUST provide a mechanism to support clients that expect an older version of a given module when the current version has had non-backwards-compatible changes.”

Requirement 4.4

OLD:

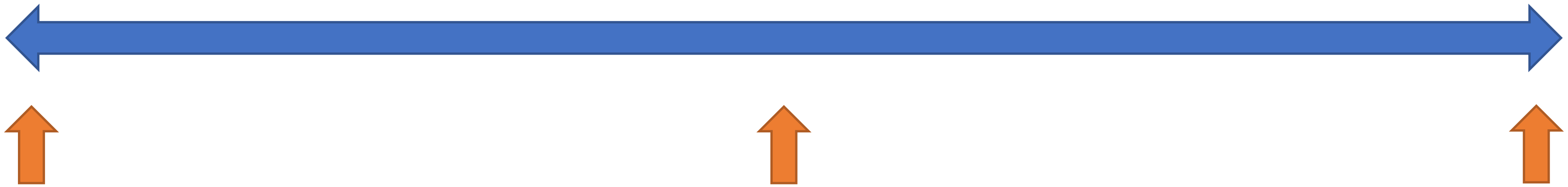
“If multiple revisions of a YANG module are published, then the solution SHOULD allow for bug fixes to be made to an older revision of the module.”

NEW:

Current Status

- The draft has been adopted as a WG document…
- …But there are questions raised during adoption

From 103…



Easy for clients
Hard for servers/authors

We are trying to find a solution somewhere in the middle

Hard for clients
Easy for servers/authors

Questions

1. **Who feels RFC7950 Section 11 is sufficient as does not require changes?**

Questions

1. **Who feels RFC7950 Section 11 is sufficient as does not require changes?**
   - The DT feels 7950 Section 11 is not sufficient
   - `Obsolete` introduces non-backwards compatibilities
   - No way to ”import module FOO at revision BAR *or later*”
   - Vendors already make non-backwards compatible changes without flagging them to users
   - Creating a brand new module *name* and not importing the old one still causes pain to uses

Questions

2. **Are non-backwards compatible changes required?**

Questions

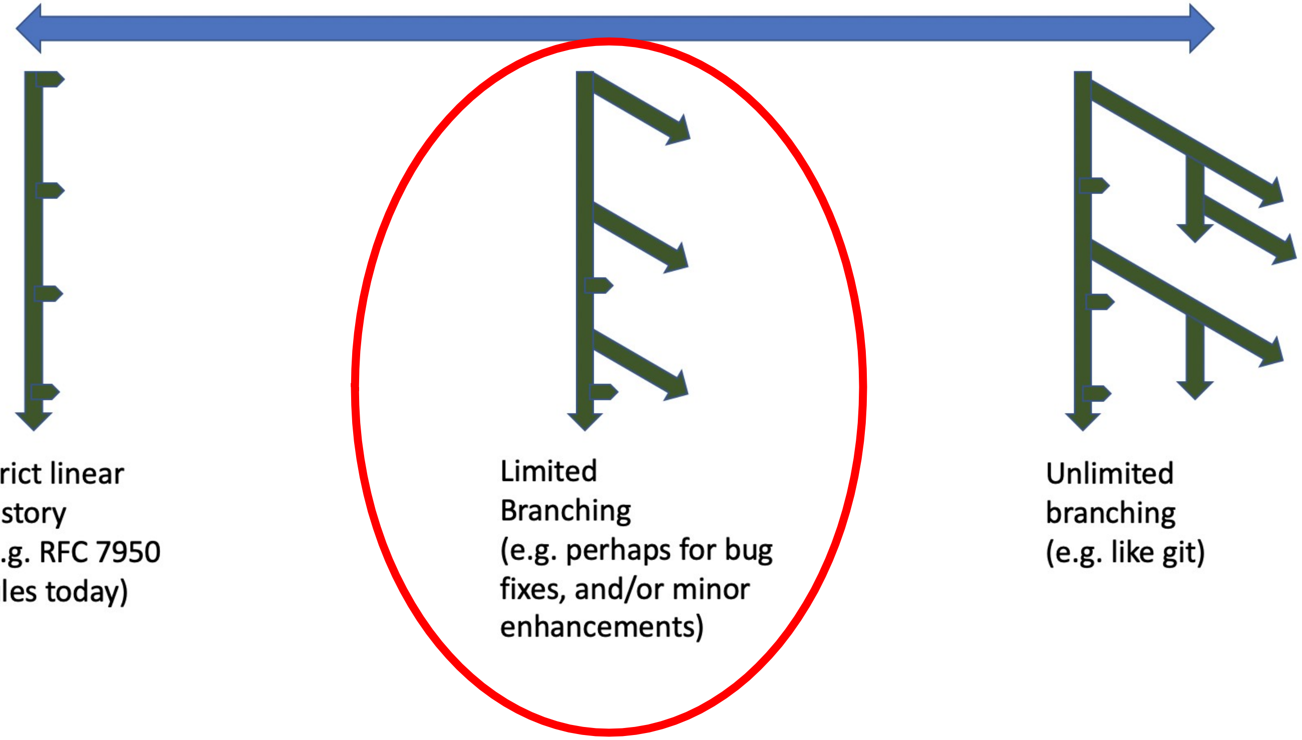
2. **Are non-backwards compatible changes required?**
    - They are undesirable, but sometimes needed
    - A bug fix may be required to a node, and it makes more sense to change it than deprecate and create a new node
    - Allowing for NBC changes in modules is being done in the already in the industry and helps to facilitate more rapid module development

Questions

3. **Is branching required? If so, how much branching is needed?**

Questions

3. **Is branching required? If so, how much branching is needed?**
   - Some branching is required to allow for bug fixes in non-current releases
   - Rob presented this last time:



Strict linear History (e.g. RFC 7950 rules today)

Limited Branching (e.g. perhaps for bug fixes, and/or minor enhancements)

Unlimited branching (e.g. like git)

Design Team update & Solution overview

NETMOD WG

March 25, 2019

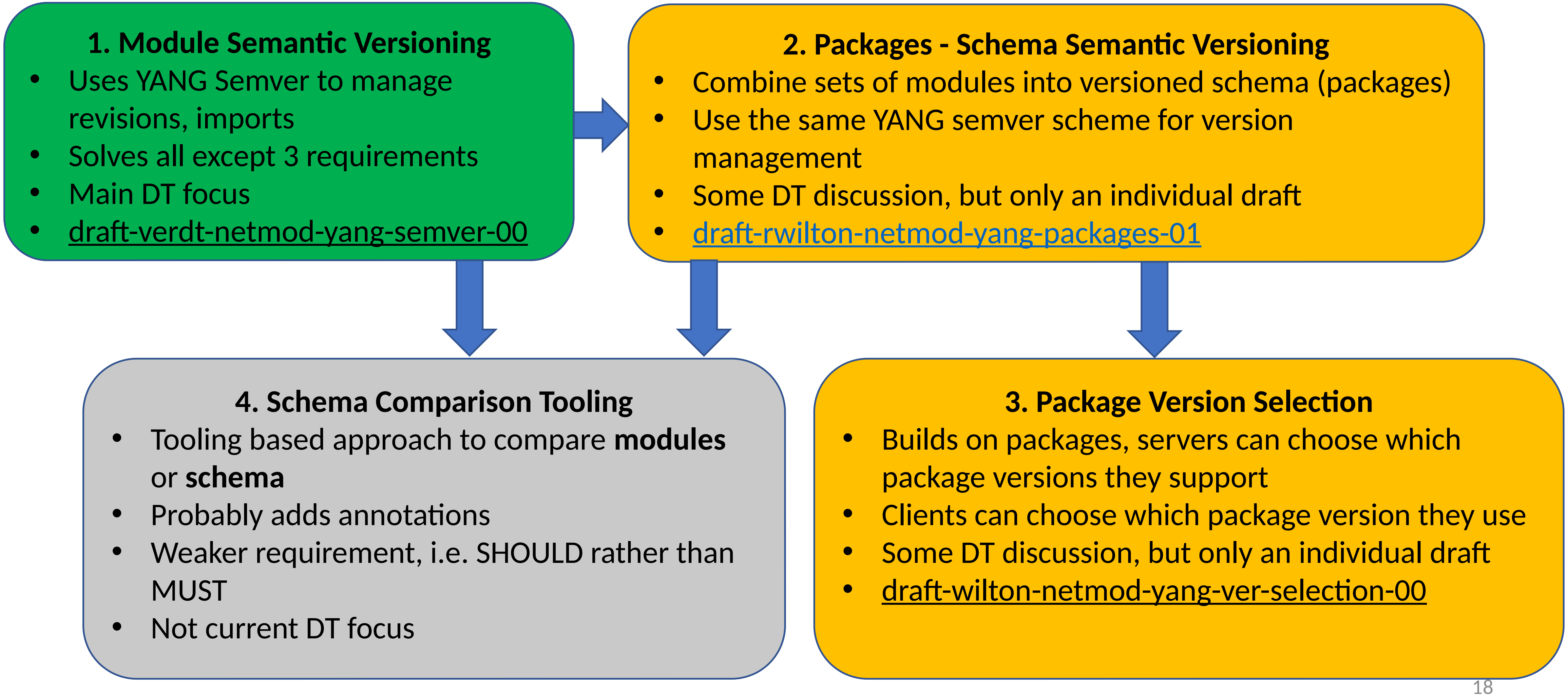
Netmod YANG Version Design Team

Rob Wilton (presenting)

Design Team Update

- Held regular weekly meetings since IETF 103
- Thank you to all members: Balazs Lengyel, Benoit Claise, Ebben Aries, Jason Sterne, Joe Clarke, Juergen Schoenwaelder, Mahesh Jethanandani, Michael (Wangzitao), Qin Wu, Reshad Rahman, Rob Wilton
- Initial focus: Reqs draft – get ready for WG adoption call
- Main focus: Module level semantic versioning draft
- Some discussion on packages and version selection (currently individual contributions, no implied DT consensus here)

YANG versioning solution – 4 parts:



1. Module Semantic Versioning

- Uses YANG Semver to manage revisions, imports
- Solves all except 3 requirements
- Main DT focus
- draft-verdt-netmod-yang-semver-00

2. Packages - Schema Semantic Versioning

- Combine sets of modules into versioned schema (packages)
- Use the same YANG semver scheme for version management
- Some DT discussion, but only an individual draft
- draft-rwilton-netmod-yang-packages-01

4. Schema Comparison Tooling

- Tooling based approach to compare **modules** or **schema**
- Probably adds annotations
- Weaker requirement, i.e. SHOULD rather than MUST
- Not current DT focus

3. Package Version Selection

- Builds on packages, servers can choose which package versions they support
- Clients can choose which package version they use
- Some DT discussion, but only an individual draft
- draft-wilton-netmod-yang-ver-selection-00

Semantic Versioning for Modules

draft-verdt-netmod-yang-semver

NETMOD WG

March 25, 2019

Netmod YANG Version Design Team
Balazs Lengyel (presenting)

Semver for Modules - Requirements

- draft-verdt-netmod-yang-versioning-reqs
- Addressed most requirements:
  - except: 2.2, 3.1, and 3.2

Semver for Modules - Solution parts

1. YANG semantic version for modules
   - (e.g. 2.4.9)
2. Semver based import-by-version
   - Allow some module versions but not all
3. Guidelines on how to use Semver
4. Status-description substatement for 'status'

5. Updates to ietf-yang-library
6. Updates to the YANG module update rules
7. Semver impacts on Yang Instance Data

Types of Changes

- Unchanged
- Editorial: semantic meaning not changed
- Backward compatible (BC)
  - RFC 7950 chapter 11 allowed changes
  - See exception below
- Non-Backward-Compatible (NBC): NBC
  - according to RFC7950,
  - status set to obsolete
- Classification helped by tools, exceptions do need manual checks
  - Pyang
  - *Future* advanced change checker
- Goal of Semver: Provide concise info about type of changes
  - avoid full comparison by tool or human

Semver:module-version

- `extension module-version      (e.g.  2.3.0)`
- Based on classification of changes
- Module name and X.Y.Z number identifies a module revision
- substatement of revision
  - revision MUST be present

```
revision 2018-02-28 {
  description "Added leaf 'wobble'";
  semver:module-version "3.1.0";
  reference “product-description ACME-Router v12”;
}
```

semantic Module-version

- `extension module-version` (e.g. 2.3.0)
- x.y.zm|M
  - x major version, stepped for NBC changes
  - y minor version, stepped for BC changes
  - z patch version stepped for editorial changes
  - m used for BC changes, needed when next “y” has already been used, and the old version MUST be updated
  - M used for NBC changes, needed when next “x” has already been used, and the old version MUST be updated
- The same x.y.z triplet MUST NOT be reused (independent of m|M
- m|M optional

YANG Semver Example

YANG semantic version numbers for an example module:

```
 0.1.0
   |
 0.2.0
   |
 1.0.0
   |  \
   |   1.1.0 -> 1.1.1m ->
1.1.2M
   |    |
   |   1.2.0 -> 1.2.1M ->
1.2.2M
   |    |
   |   1.3.0 -> 1.3.1
   |
 2.0.0
   |
 3.0.0
      \
       3.1.0
```

- 0.1.0 - first beta module version
- 0.2.0 - second beta module version (with NBC changes)
- 1.0.0 - first release (may have NBC changes from 0.2.0)
- 1.1.0 - added new functionality, leaf "foo" (BC)
- 1.2.0 - added new functionality, leaf "baz" (BC)
- 1.3.0 - improve existing functionality, added leaf "foo-64" (BC)
- 1.3.1 - improve description wording for "foo-64" (Editorial)
- 1.1.1m - backport "foo-64" leaf to 1.1.x (BC)
- 2.0.0 - change existing model for performance reasons, e.g. re-key list (NBC)
- 1.1.2M - NBC bug fix, not required in 2.0.0 due to model changes (NBC)
- 3.0.0 - NBC bugfix, rename "baz" to "bar"; also add new BC leaf "wibble"; (NBC)
- 1.2.1M - backport NBC fix, changing "baz" to "bar"
- 1.2.2M - backport "wibble". This is a BC change but "M" modifier is sticky.
- 3.1.0 - introduce new leaf "wobble" (BC)

Why this numbering scheme ?

- Based on semver.org 2.0.0
  - Well know in industry
- Allows bugfix on old tracks
  - both BC and NBC
- Compromise between powerful & simple
  - Encourage “mostly” single track development m|M modifiers the exception
  - supports limited updates to older versions
- Sufficient for SDOs and vendors
  - vendors have greater need for semver

Import by version

- We want to allow some versions, but not all and more than a single
- semver:version extension as substatement to import
  - exact version `semver:version 1.1.2;`
  - greater then semver `semver:version 1.2.0+;`
  - range of semver `semver:version 1.0.0-1.MAX.MAX;`
- Multiple statements are “Or”-ed
  - any version allowed by one of the statements is acceptable for import
- Assumption: NBC changes usually do not harm import
- Solution is backwards compatible
  - unchanged import-by-revision takes precedence

Import by version

Available versions:

```
0.1.0, 1.0.0, 1.1.0, 1.1.1m, 1.1.2M, 1.2.0, 1.2.1M,
1.3.0, 1.3.1, 2.0.0, 2.1.0
```

```
import example-module {
     semver:version 1.1.0-1.1.1;
     semver:version 1.2.0;
     semver:version 1.3.0+;
   }
```

Acceptable versions:

```
1.1.0, 1.1.1m, 1.2.0, 1.3.0, 1.3.1, 2.0.0, 2.1.0
```

How to use YANG Semver

- Guidelines to YANG model authors
  - Minimize NBC changes
  - Reuse of semver number MUST NOT be done
  - Try to develop on the main branch. Avoid m|M modifiers
  - m|M sticky, once used they stay till major/minor number is changed
- When doing NBC changes
  - Try to lessen the pain
  - Try to deprecate before obsoleting or removing a node
  - Use status-description to indicate replacement
  - Try to support old AND new nodes side-by-side

Updates to ietf-yang-library

- Add version to modules
- Resolving ambiguous module imports
  - multiple module versions may be available for import
    - may result in the presence/absence of data nodes (uses/grouping)
    - may result in different valuespace for leafs
  - use implemented or or gratest semantic-version or latest revision
- Indicate whether deprecated/obsolete nodes are implemented or not

Semver for Modules - Way forward

- Asking the chairs, audience:
  - We request adoption by the WG ?
  - This is not a full solution yet – we will still have debates !

- Open Issues
  - Documented at Ver-Dt-Github
  - Please add your comments

Semantic versioning for YANG schema

draft-rwilton-netmod-yang-packages-01

NETMOD WG

March 25, 2019

Rob Wilton (presenting)

I E T F

YANG Package - Overview

Defines a versioned YANG schema as the set of YANG module versions/revisions

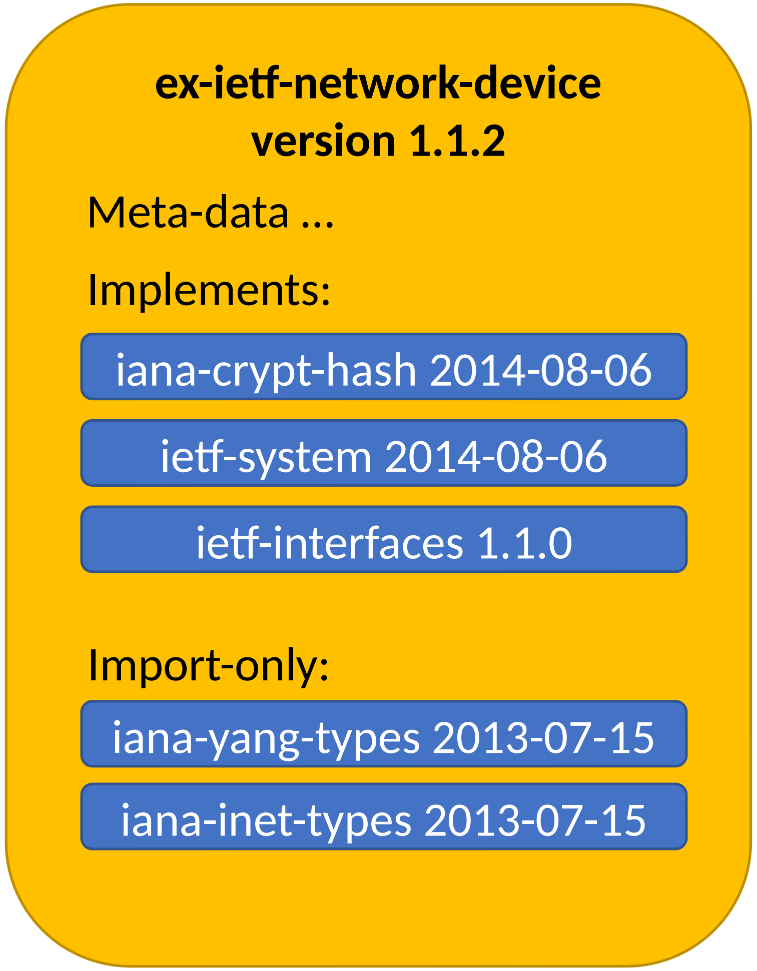
- Uses the same module YANG semver scheme
- Hierarchical - packages can import other packages
- Available offline in a YANG instance data file
- Available on the device (e.g. in YANG library)

YANG Package – Why?

Aim to solve several problems:

- Need to version sets of modules instead of just individual modules
- Encourage more consistency in implementations
- Some schema contain 100+ modules, managing these as a flat list is unwieldy
- To avoid downloading/check the full module list from a device. Making the schema available offline, and then check that the device is using (or just compatible with) the expected schema

Example Package - **ex-ietf-network-device**



ex-ietf-network-device
version 1.1.2

Meta-data …

Implements:

- iana-crypt-hash 2014-08-06
- ietf-system 2014-08-06
- ietf-interfaces 1.1.0

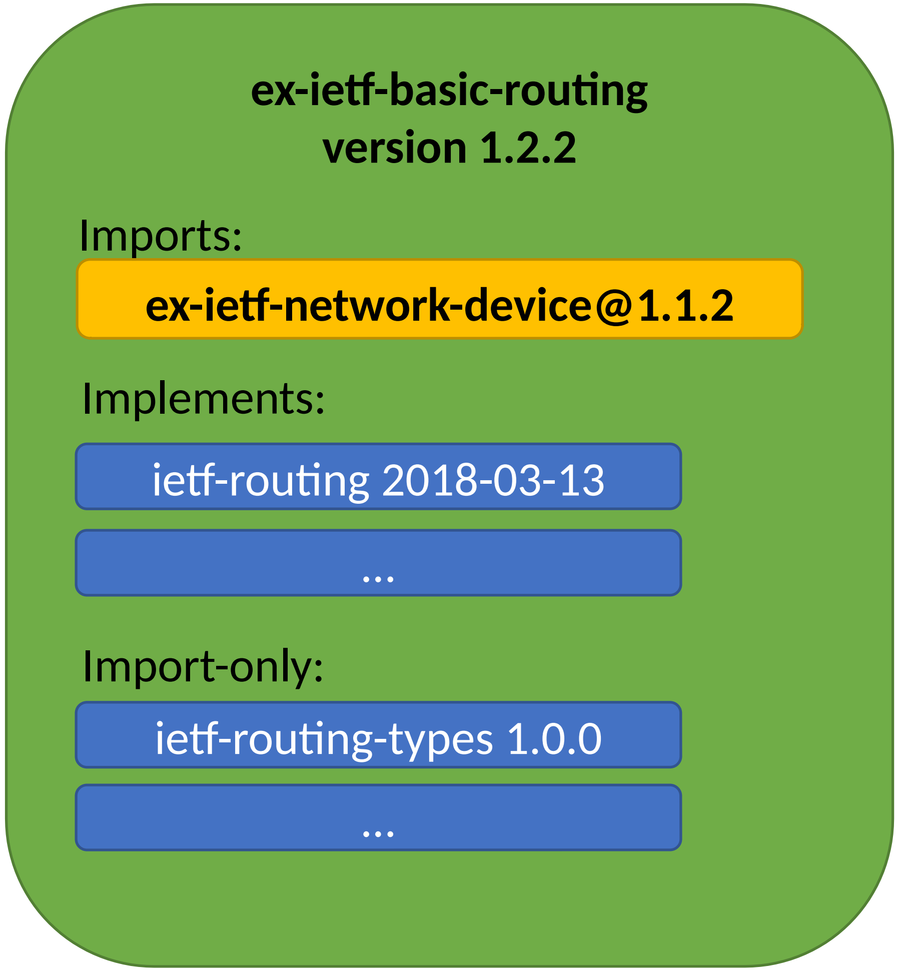
Import-only:

- iana-yang-types 2013-07-15
- iana-inet-types 2013-07-15

Definition includes:

- Metadata:
  - URLs to find package/module definitions
  - Mandatory features
- Imported packages
- Implemented module version/revisions
- Import-only module versions/revisions
- Import conflict resolution

Example Package 2 - **example-ietf-basic-routing**



ex-ietf-basic-routing
version 1.2.2

Imports:

ex-ietf-network-device@1.1.2

Implements:

ietf-routing 2018-03-13

…

Import-only:

ietf-routing-types 1.0.0

…

- Ex-ietf-basic-routing imports ex-ex-ietf-network-device and defines more module versions
- Any version conflict/change must be explicitly resolved
- Package version indicates nature of changes in the modules or package import

Schema version selection

draft-wilton-netmod-yang-ver-selection-00

NETMOD WG

March 25, 2019

Reshad Rahman (presenting), Rob Wilton

I E T F

YANG version selection - Goals

- To address requirements 3.1 and 3.2 from draft-verdt-netmod-yang-versioning-reqs-02:

```
3. Requirements related to supporting existing clients in a
backwards-compatible way:

   3.1 The solution MUST provide a mechanism to allow servers to
   support existing clients in a backwards-compatible way.

   3.2 The solution MUST provide a mechanism to support clients
   that expect an older version of a given module when the
   current version has had non-backwards-compatible changes.
```

- Enables servers to support separate schema families (e.g. IETF, OpenConfig, Native) and allows clients to choose a schema family

YANG Version Selection - Summary

- Allows servers to do NBC changes without forcing clients to immediately migrate to the new module versions
- Makes use of YANG packages defined in draft-rwilton-netmod-yang-packages
- Provides a mechanism for servers to advertise support for multiple versions of YANG packages
- Allows clients to choose, among the ones advertised by the server, which YANG package version they use

YANG Version Selection – Summary (2)

- Servers are NOT required to concurrently support clients using different schema versions
- Servers are NOT required to support every published version of a YANG package
- Servers are NOT required to support all parts of all versioned schema. E.g. for some NBC changes, it may not be possible for the server to support both the old and new versions

YANG Version Selection – Overview

- A versioned schema is a YANG schema with an associated YANG semantic version number, e.g. a YANG package
- A schema-set is a set of related versioned YANG schema, one for each supported datastore
- Servers support configuration for the default schema-set version to use for default NETCONF/RESTCONF connections
- Servers support configuration for secondary NETCONF/RESTCONF instances (using different port numbers) which use a different schema-set version
- Clients choose which schema-set they use by using the corresponding NETCONF/RESTCONF port number

YANG Version Selection – Data Tree

```
module: ietf-schema-version-selection
    +--rw schema-selection
    |  +--rw schema-sets* [name]
    |     +--rw name string
    |     +--rw netconf! {secondary-schema-set}?
    |     |  +--rw port? inet:port-number
    |     +--rw restconf! {secondary-schema-set}?
    |     |  +--rw port? inet:port-number
    |     |  +--rw root-path? inet:uri
    |     +--ro datastores* [datastore]
    |        +--ro datastore ds:datastore-ref
    |        +--ro package
    |           +--ro name?
    |           | -> /yanglib:yang-library/pkg:package/name
    |           +--ro version? Leafref
    +--rw default-schema-set?
            -> /schema-selection/schema-sets/name
            {default-schema-set}?
```

NETMOD Versioning Design Team - Next Steps

- WG adoption of draft-verdt-netmod-yang-semver-00 ?
- Does WG agree with the current direction of the solution, e.g. for the DT to work on packages and version selection drafts.

draft-ietf-core-yang-cbor-07

Backup Slides - Requirements

Requirement 1.4

OLD:

“The solution MUST allow for modules to be versioned by software release. In particular, backwards-compatible enhancements and bug fixes MUST be allowed in any non-latest release.”

NEW:

“The solution MUST allow for backwards-compatible enhancements and bug fixes, as well as non-backwards-compatible bug fixes in non-latest-release modules.”

Requirement 3.2

OLD:

“The solution MUST provide a mechanism to allow servers to simultaneously support clients using different revisions of modules. A client's choice of particular revision of one or more modules may restrict the particular revision of other modules that may be used in the same request or session.”

NEW:

“The solution MUST provide a mechanism to support clients that expect an older version of a given module when the current version has had non-backwards-compatible changes.”

Requirement 4.4

OLD:

“If multiple revisions of a YANG module are published, then the solution SHOULD allow for bug fixes to be made to an older revision of the module.”

NEW:

Backup Slides – Module Semver

Status-description

- Defined
```
extension semver:status-description
```
- Freeform text substatement to status
- Inform user about:
  - replacement node
  - time to obsoleting/removal of deprecated nodes
  - reason for status change
  - etc.

Semver and Yang Instance Data

- Instance data is not versioned
  - Backward compatibility is not defined
- Target YANG module's version is available for Instance data
  - Helps to determine whether instance data is usable with updated versions of a module
- Updating/Upgrading instance data after a server upgrade involving module updates – out of scope