YANG Versioning Design Team Update
YANG Module Versioning Requirements Draft

NETMOD - IETF 102

draft-verdt-netmod-yang-versioning-reqs-00

Robert Wilton, Cisco
(Presenting, on behalf of design team)

YANG Versioning Design Team Update

- Design team formed after IETF 101
- We have held weekly meetings for the last couple of months
- Focus has been on Problem Stmt and Reqs
- We have produced a requirements draft (will cover next)
- Some discussion needed on next steps (will cover at the end)

Requirements Draft

- Short summary of the problem
  - Also includes and extends on the example problems from clacla-netmod-yang-model-update
- List of requirements
- WG adoption

Problem statement - Summary

YANG Versioning:

- Want to allow YANG models to evolve rather than requiring perfection on day 1:
  - Current mechanism – once published, no non backwards compatible changes. Too inflexible.
  - Non backwards compatible changes require either a new module name or new path. Both of which have high impact on implementations, importing modules.

Problem statement – Related issues

- Ambiguity to clients whether “*status deprecated*” nodes are implemented
- YANG lacks the ability to add detail about data node lifecycle
- Some systems (e.g where YANG modules are generated) require support for allowing non backwards compatible updates to YANG modules

Requirements (1)
Non backwards Compatible updates:

1.1 A mechanism is REQUIRED to update a module in a non-backward compatible way without forcing all modules with import dependencies on the updated module from being updated at the same time (e.g. to change its import to use a new module name).

1.2 A mechanism is REQUIRED to update a module in a non-backward compatible way without forcing all clients/servers to access data nodes in the model on new paths, or in a new module namespace. Specifically, if a particular data node is updated in a non-backward compatible way then it may be desirable for it to be available on the same path and in the same module namespace.

1.3 A refined form of YANG's 'import' statement MUST be provided that is more restrictive than "import any revision" and less restrictive than "import a specific revision". Once non-backward compatible changes to modules are allowed, the refined import statement is used to express the correct dependency between modules.

Requirements (2) Identifying module revisions:

2.1 Readers of modules, and tools that use modules, MUST be able to determine whether changes between two revisions of a module constitute a backward compatible or non-backward compatible version change. In addition, it MAY be helpful to identify whether changes represent bug fixes, new functionality, or both.

2.2 A mechanism SHOULD be defined to determine whether data nodes between two arbitrary YANG module revisions have (i) not changed, (ii) changed in a backward compatible way, (iii) changed in a non-backward compatible way.

Requirements (3)
Supporting existing clients:

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

       3.2  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.
```

Requirements (4)
Documenting data node life cycle:

4.1 A mechanism is REQUIRED to allow a client to
determine
whether deprecated nodes are implemented by the
server.

4.2 If a data node is deprecated or obsolete then it MUST
be
possible to document in the YANG module what
alternatives
exist, the reason for the status change, or any other
status
related information.

4.3 A mechanism is REQUIRED to indicate that certain
definitions
in a YANG module will become status obsolete in
future
revisions but definitions marked as such MUST still
be
implemented by compliant servers

Requirements (5) Documentation and education:

5.1 The solution MUST provide guidance to model authors and clients on how to use the new YANG versioning scheme.

5.2 The solution is REQUIRED to describe how to transition from the existing YANG 1.0/1.1 versioning scheme to the new scheme.

5.3 The solution MUST describe how the versioning scheme affects the interpretation of instance data and references to instance data, for which the schema definition has been updated in a non backward compatible way.

Requirements Draft – Next steps

- Does the WG agree with the requirements?
- DT aim is for the requirements draft to be adopted by the WG
- Whether this is as a transient work item, or progressed to an RFC, is up to WG chairs & WG
  - DT opinion is that this depends on what shape the solution draft(s) take.

DT Solutions - Next Steps

1. Consider possible solutions:
   - Solution proposals/ideas are welcome
   - Only currently aware of one solution draft: further work required to address all requirements
   - Update on clacla-netmod-yang-model-update solution draft (time permitting)
2. Progress solution to a WG adopted solution draft(s)

DT – Next steps, potential issue

- The solution is likely to:
  - Add semantic version number to YANG modules
  - Add import by version numbers/ranges
  - Change meaning of “status: deprecated”
  - Change module update rules
  - Perhaps require version selection
- Will need to update RFC 7950
- Also likely extensions to: NETCONF, RESTCONF, YANG library

Potential issue – So how do we do this work?

- DT hasn’t reached consensus on this:
  - Majority prefer doing a separate solution specific RFC(s) that will update 7950.
  - Or does this need to be YANG-bis?
- Or perhaps the two efforts can be separated and done in parallel?

Thanks for listening