YANG Semantic Versioning

draft-ietf-netmod-yang-semver

NETMOD WG

July 2020

Presentation Authors: Jason Sterne & Joe Clarke
Presenting: Joe Clarke

IETF

Changes since March 2020 virtual interim

- New suffixes to replace ‘m’ and ‘M’
- Definition of revision-label-scheme identity for YANG semver
- Guidelines on how to apply YANG semver revision-labels during module development
- Expansion of version component size (from 32768 to 2^31 - 1)
- Example IETF module development flow (see Appendix A)

New Suffixes

- Gone are ‘m’ and ‘M’ to be the sticky “backwards-compatible” and “non-backwards-compatible” tags
- Instead, “_compatible” and “_non_compatible” are used
- Yes, they are longer, but their meaning is clearer and these sticky version tags should be avoided when possible
- Example use:

```
1.0.0
   |
1.1.0 – 1.1.1 – 1.1.2_compatible – 1.1.3_non_compatible
   |
2.0.0
```

New feature added based on customer demand

Breaking change made based on customer demand

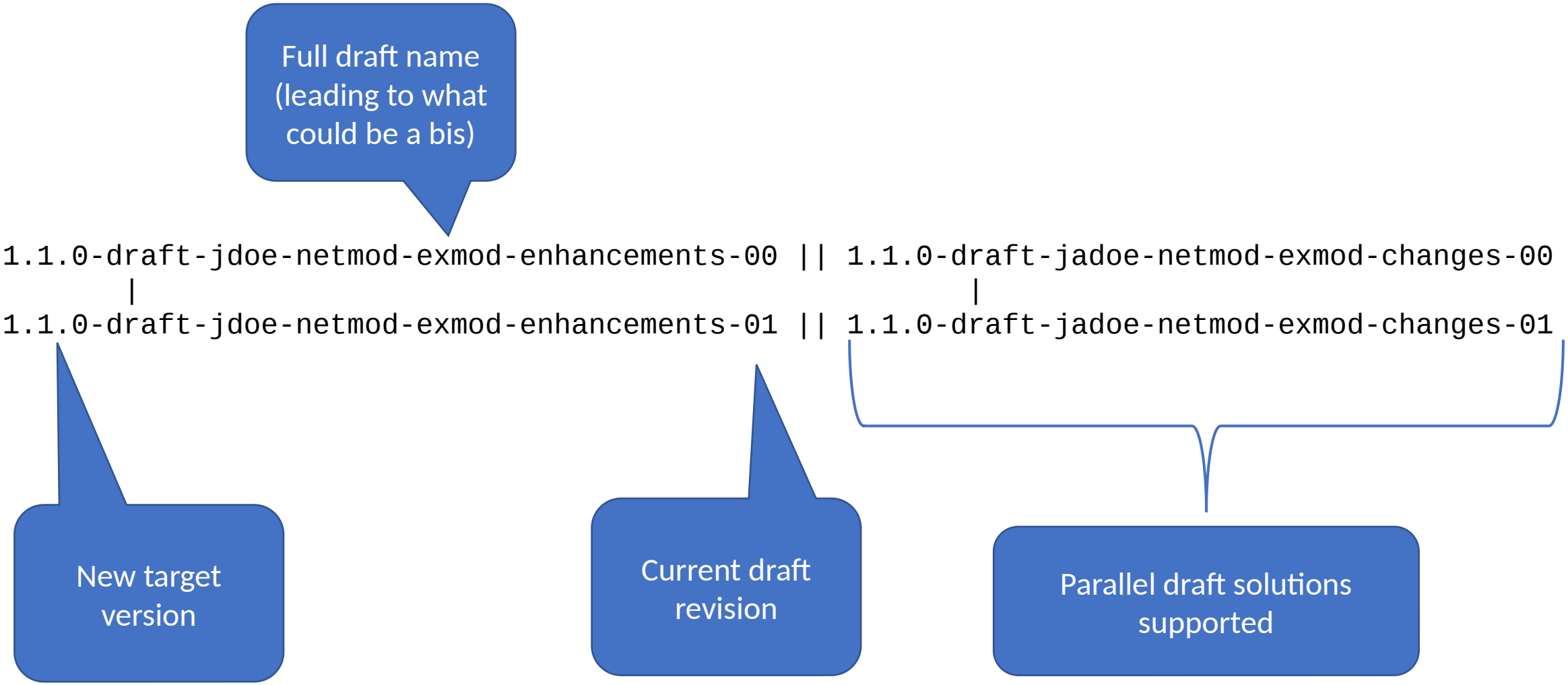
Revision-label scheme for YANG Semver

```
identity yang-semver {
     base rev:revision-label-scheme-base-identity;
     description
       "The revision-label scheme corresponds to the YANG semver scheme
        which is defined by the pattern in the 'version' typedef below.
        The rules governing this revision-label scheme are defined in the
        reference for this identity.";
     reference
       "RFC XXXX: YANG Semantic Versioning.";
   }
```

Guidelines for YANG Semver for Development

```
1.0.0-draft-ietf-netmod-example-module-00
  |
1.0.0-draft-ietf-netmod-example-module-01
  |
1.0.0-draft-ietf-netmod-example-module-02
```

And Subsequent Changes To A Module



Discussion Topics

Pre-release version precedence

GitHub Issue #60: If the MAJOR, MINOR, PATCH of a pre-release version doesn't change or jumps forward, the pre-release precedence rules are well-established based on semver.org. However, if a module is initially going to be a MAJOR version ahead but during development it's reverted to only a MINOR version increment, the precedence rules break. For example:

1.0.0 < 2.0.0-pre-release <broken!> 1.1.0-pre-release < 1.1.0

The above could be fixed simply by saying pre-releases don't have implied precedence and the module lineage should be used instead. But then we lose any semantic meaning in the revision-labels. Though, with the draft names in the pre-release strings, there is a direct link back to the draft, which has history in Data Tracker.

Another thought is to say that revision-labels can never go backwards. That if you decide on 2.0.0 as the next string, that the final product will be 2.0.0. This can work because it will most likely trigger authors to use the next MINOR (or even PATCH) version throughout development until the final release. Then, and only then is a true YANG semver assigned based on the totality of the development.

YANG Semver mandatory for IETF modules

GitHub Issue #45: Should all newly published IETF YANG modules include a yang-semver revision-label?

Authors' proposal: yes

- helps readers understand "at a glance" if two revisions of a module are backwards compatible

Should IANA controlled YANG modules (e.g. BGP address families, RFC8294 routing types, etc) include a yang-semver revision-label (and nbc statement)? The rules would be fairly simple for these IANA modules.

Next Steps

YANG Semver – Next steps

- Resolve last 2 outstanding issues
- Take to WG LC

Backup