NETMOD
Internet-Draft
Intended status: Informational
Expires: 19 June 2026

R. Wilton
Cisco

16 December 2025

Guidance for Managing YANG Modules in RFCs and IANA Registries
draft-verdt-iana-yang-guidance-00

Abstract

This document provides guidance to the RFC Editor and IANA on managing YANG modules in RFCs and IANA registries, ensuring consistent application of YANG Semantic Versioning rules.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://rgwilton.github.io/iana-yang-guidance/draft-verdt-iana-yang-guidance.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-verdt-iana-yang-guidance/.

Discussion of this document takes place on the Network Modelling Working Group mailing list (mailto:netmod@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/netmod/. Subscribe at https://www.ietf.org/mailman/listinfo/netmod/.

Source for this draft and an issue tracker can be found at https://github.com/rgwilton/iana-yang-guidance.

Status of This Memo

Table of Contents

1. Open Issues/Questions:

1. Do we need guidance to IANA in this document to list modules both by revision date and version?

2. This document is informational, is it appropriate to use RFC 2119 language?

2. Introduction

YANG [RFC7950] modules are used to model network management data and protocols. The IETF publishes YANG modules as part of RFCs, and the Internet Assigned Numbers Authority (IANA) maintains YANG modules that are derived from IANA registries. Both processes require careful attention to module versioning to ensure that implementations can correctly assess compatibility when modules are updated.

This document provides informational guidance to both the RFC Editor and IANA for managing YANG modules in two distinct scenarios:

1. *Managing YANG Modules in RFCs*: When documents containing YANG modules are approved by the IESG and processed for publication as RFCs, both the RFC Editor and IANA have responsibilities to ensure that modules are correctly versioned and published.

2. *Managing IANA-Maintained YANG Modules*: When IANA registries are updated, any YANG modules derived from those registries must be updated accordingly with proper versioning.

The guidance in this document is informational rather than prescriptive. It describes recommended practices and procedures that reflect current consensus within the NETMOD working group and the IETF operations and management community. While following this guidance will help ensure consistent and correct handling of YANG modules, specific situations may require consultation with experts (as described in Section 7).

> *Note:* In addition to the guidance detailed in this document, there is a broader, ongoing discussion within the IETF community around the processes and responsibilities for managing YANG modules in RFCs. For further information and the latest proposals, see [I-D.boucadair-veloce-yang]. The recommendations and operational practices described here may be revised in the future to reflect outcomes from that work.

The procedures and classifications in this document are based on the following IETF specifications:

* [I-D.ietf-netmod-yang-module-versioning] - Defines updated YANG module revision handling, including rules for backwards-compatible and non-backwards-compatible changes

* [I-D.ietf-netmod-yang-semver] - Defines YANG Semantic Versioning (YANG Semver) for YANG modules

* [I-D.ietf-netmod-yang-module-filename] - Defines filename conventions for YANG modules versioned using YANG Semver

* [I-D.ietf-netmod-rfc8407bis] - Provides general guidelines for IETF YANG module authors

3. Conventions and Definitions

This document uses the following terminology from [I-D.ietf-netmod-yang-module-versioning]:

Backwards-Compatible (BC) Change A change to a YANG module that conforms to the backwards-compatible update rules defined in section 3.1.1 of [I-D.ietf-netmod-yang-module-versioning]. BC changes require incrementing the MINOR version number.

Non-Backwards-Compatible (NBC) Change A change to a YANG module that does not conform to the backwards-compatible update rules defined in section 3.1.2 of [I-D.ietf-netmod-yang-module-versioning]. NBC changes require incrementing the MAJOR version number and adding the rev:non-backwards-compatible extension statement within the revision statement in the YANG module.

This document uses the following terminology from [I-D.ietf-netmod-yang-semver]:

YANG Semver YANG Semantic Versioning - a version identifier in the format _MAJOR.MINOR.PATCH_COMPAT_ that indicates the compatibility level of a YANG module, as defined in [I-D.ietf-netmod-yang-semver].

Editorial Change A change to a YANG module that does not affect the semantic meaning or functionality of the module. Editorial changes only require incrementing the PATCH version number.

In addition, this document defines:

IANA-Maintained Module A YANG module maintained by IANA, typically derived from one or more IANA registries. These modules have names starting with "iana-" (e.g., iana-if-type, iana-routing-types).

4. Background on YANG Versioning

4.1. YANG Semantic Versioning

YANG Semantic Versioning (YANG Semver), defined in [I-D.ietf-netmod-yang-semver], uses a version identifier in the format MAJOR.MINOR.PATCH (with an optional _COMPAT suffix for branched development):

* *MAJOR* version increments indicate non-backwards-compatible (NBC) changes
* *MINOR* version increments indicate backwards-compatible (BC) feature additions
* *PATCH* version increments indicate editorial or documentation-only changes
* *_COMPAT* is used for branched development trees and is not applicable to modules published by the RFC Editor in RFCs that are expected to follow a linear development, (*TODO, but may be useful/needed if we allow verified errata against YANG modules*) or maintained by IANA.

For example, if a published IETF YANG module is at version 1.2.3: - A non-backwards-compatible change would update it to 2.0.0 - A backwards-compatible feature addition would update it to 1.3.0 - An editorial change would update it to 1.2.4

Pre-release versions (versions with MAJOR = 0 or with a pre-release suffix such as "-draft-verdt-iana-yang-guidance") indicate modules that have not completed the IETF standardization process and whose revision content is subject to change in non-backwards-compatible ways without corresponding changes to the major version number.

4.2. Backwards Compatibility Rules

The rules that determine whether a change to a YANG module is backwards-compatible or non-backwards-compatible are defined in Section 3.1 of [I-D.ietf-netmod-yang-module-versioning]. These rules refine and extend the update rules specified in Section 11 of [RFC7950].

Section 3.1.1 of [I-D.ietf-netmod-yang-module-versioning] defines backwards-compatible changes, which include:

* Adding new schema nodes (e.g., new enum values, identities, leafs, containers)
* Adding new optional features or extensions

* Changing the status of a schema node from "current" to "deprecated"

* Adding or updating "description" and "reference" statements (provided the semantic meaning is unchanged)

* Expanding constraints (e.g., widening ranges, adding enum values)

Section 3.1.2 of [I-D.ietf-netmod-yang-module-versioning] defines non-backwards-compatible changes, which include:

* Removing schema nodes (unless they already have status "obsolete")

* Changing the status of a schema node from "current" or "deprecated" to "obsolete"

* Renaming schema nodes or changing their identifiers

* Changing data types in ways that alter syntax or semantics

* Changing numeric values assigned to enumerations

* Restricting constraints (e.g., narrowing ranges, removing enum values)

* Modifying "description" statements in ways that change semantic meaning or behavior

In addition, section 4.4 of [I-D.ietf-netmod-yang-semver] defines editorial changes as the subset of backwards-compatible changes that have no impact on the semantics or syntax of a YANG module, which include:

* Corrections to comments, descriptions, or references that do not change the semantic meaning

* Formatting improvements such as whitespace or indentation changes

* Corrections to typographical errors in description text

* Updates to contact information or copyright statements

* Changes to import statements that do not affect module functionality (e.g., updating import prefixes for readability)

4.3. The rev:non-backwards-compatible Extension

The YANG module versioning framework [I-D.ietf-netmod-yang-module-versioning] defines the "rev:non-backwards-compatible" extension statement. This extension MUST be added as a substatement of a revision statement whenever that revision contains non-backwards-compatible changes relative to the previous revision.

The following example illustrates this extension in use. In the example, an identity 'foo' was added in version 1.3.0, but was subsequently renamed to 'bar' in version 2.0.0. Since renaming is a non-backwards-compatible change, the major version number is incremented and the rev:non-backwards-compatible extension is included in the revision statement in version 2.0.0 of the YANG module:

```
  revision 2025-11-15 {
    ysv:version "2.0.0";
    rev:non-backwards-compatible;
    description
      "Renamed identity 'foo' to 'bar'.";
  }
  revision 2025-06-01 {
    ysv:version "1.3.0";
    description
      "Added identity 'foo'.";
  }
  ...
```

Figure 1: Revision history example from a YANG module

4.4. Module Immutability

A fundamental principle of YANG module versioning is that once a module revision is published with a specific revision date and version number, its content is immutable (much like an RFC is). The published content of that revision MUST NOT change. Any change to the module content requires publishing a new revision with a new revision date and an updated YANG Semver.

This immutability principle has important implications:

* Modules in Internet-Drafts SHOULD use pre-release versions (e.g., 0.1.0 or 2.0.0-draft) to indicate that the content may still change

* Once a document is approved by the IESG, the module version MUST be updated to a release version (e.g., 1.0.0, or 2.0.0) before publication as an RFC.

* IANA-maintained modules MUST publish a new revision any time the registry changes require module updates

5. YANG Modules in Documents Being Published as RFCs

This section describes the workflow and responsibilities for managing YANG modules in documents that have been approved by the IESG and are being processed for publication as RFCs. Both the RFC Editor and IANA have roles in this process.

5.1. Core Requirements

All YANG modules published by the RFC Editor or maintained by IANA MUST meet the following requirements:

1. *YANG Semver Version*: Every module MUST include a semantic version number using the ysv:version statement in its most recent revision. The version MUST be correct relative to any previously published version of the same module (either in a previous RFC or on the IANA website).

2. *NBC Extension for NBC Changes*: If the module contains non-backwards-compatible changes relative to the previously published version, the revision statement MUST include the rev:non-backwards-compatible extension.

3. *Revision Immutability*: A published YANG module with a specific revision date and version number is immutable. Its content MUST NOT change without also changing the revision date and version number. For this reason, modules in Internet-Drafts SHOULD use pre-release versions (e.g., versions with MAJOR = 0 such as 0.1.0, or versions with a pre-release suffix such as 2.0.0-draft) to indicate that content may still change before final publication.

4. *RFC Code Markers*: YANG modules in RFCs MUST be properly marked with <CODE BEGINS> and <CODE ENDS> markers (or equivalent in the source format) to enable automated extraction. The markers MUST include the filename following the conventions in [I-D.ietf-netmod-yang-module-filename].

5.2. Workflow Steps

The following steps describe the coordinated process between the RFC Editor and IANA for handling YANG modules during RFC publication:

5.2.1. Step 1: IESG Approval with Pre-Release Version

When a document is approved by the IESG, any YANG modules it contains typically have pre-release version numbers (e.g., 0.1.0 or 1.0.0-draft). These pre-release versions indicate that the module content may still be subject to editorial changes during RFC Editor processing.

5.2.2. Step 2: RFC Editor Processing

During RFC Editor processing, the RFC Editor may make editorial changes to the YANG module, such as:

* Improving description text for clarity without changing semantic meaning
* Updating references to use final RFC numbers instead of draft names
* Correcting typographical errors
* Standardizing formatting and style

These editorial changes are appropriate and expected. The RFC Editor SHOULD:

* Coordinate with document authors regarding any substantive changes
* Ensure that only editorial changes (as defined in Section 4) are made without author consultation
* If more significant changes are needed that might be backwards-compatible or non-backwards-compatible, consult with the authors to determine the correct version number and whether the rev:non-backwards-compatible extension is required

5.2.3. Step 3: Finalizing the Module Version

Before publication, the module version MUST be updated from the pre-release version to a release version. The RFC Editor, in coordination with the document authors:

* Updates the version to remove pre-release indicators (e.g., 0.1.0 → 1.0.0, or 1.0.0-draft → 1.0.0)

* Ensures the version correctly reflects the relationship to any previously published version of the module

* Adds the rev:non-backwards-compatible extension if NBC changes have occurred since the previous publication

* Updates the revision date to reflect the date of the final revision

5.2.4. Step 4: IANA Delay of Publication

IANA SHOULD delay publishing the YANG module to the IANA YANG Parameters registry until the RFC Editor has completed editing the module. This coordination ensures that:

* The IANA-published version matches the RFC-published version exactly

* No discrepancies exist between the two authoritative sources

* The module reference to the RFC (if present) is correct

5.2.5. Step 5: Coordinated Publication

Once the RFC Editor has finalized the module:

* The RFC is published with the final module content

* IANA publishes the module to the IANA YANG Parameters registry at approximately the same time

* The module filename follows the conventions in [I-D.ietf-netmod-yang-module-filename]

* IANA registers the module in the "YANG Module Names" registry if it is not already registered

5.3. Determining the Correct Version

The correct version for a module in an RFC depends on its relationship to any previously published version:

* *New Module* (never published before): Version 1.0.0 is typically appropriate for the first publication

* *Updated Module* (updating a module from a previous RFC): The version increment depends on the nature of changes:
  - Editorial changes only → PATCH increment (e.g., 1.0.0 → 1.0.1)
  - Backwards-compatible additions → MINOR increment (e.g., 1.0.0 → 1.1.0)
  - Non-backwards-compatible changes → MAJOR increment (e.g., 1.0.0 → 2.0.0) with NBC extension

Tooling (described in Appendix A) can assist in determining the correct version by comparing the new module with the previously published version. The pyang --check-update-from command is particularly useful for detecting NBC changes.

However, tools have limitations and cannot always detect editorial versus backwards-compatible changes, particularly in description text and must/when expressions. In such cases:

* Review the classification guidance in Appendix B
* Consult with document authors who understand the intent of changes
* Seek expert guidance as described in Section 7 if uncertainty remains
* When in doubt, choose the more conservative classification (e.g., NBC rather than BC, or BC rather than editorial) to better highlight potential risks to implementations

5.4. Verification by IANA

When registering a YANG module from an RFC, IANA SHOULD verify:

1. The module includes a ysv:version statement in the most recent revision
2. If the MAJOR version has incremented from a previous publication, the rev:non-backwards-compatible extension is present
3. The module filename follows the conventions in [I-D.ietf-netmod-yang-module-filename]
4. The module is syntactically valid (can be validated using tools described in Appendix A)

If issues are found, IANA SHOULD report them to the RFC Editor, the document authors, and the NETMOD working group.

6. IANA-Maintained YANG Modules

This section describes the process for IANA to update and publish YANG modules that are maintained by IANA and derived from IANA registries.

6.1. Overview

Many IANA registries have corresponding YANG modules that represent registry contents in a machine-readable format. Examples include:

* *iana-if-type* - derived from the Interface Types (ifType) registry
* *iana-routing-types* - derived from Address Family Numbers and SAFI Parameters registries
* *iana-bgp-types* - derived from BGP Parameters registries

When these registries are updated, the corresponding YANG modules MUST be updated accordingly, following the same versioning rules described in Section 4.

6.2. Characteristics of IANA-Maintained Modules

IANA-maintained YANG modules typically:

* Have names starting with "iana-"
* Contain primarily enumeration typedefs or identity definitions that map directly to registry entries
* Are updated more frequently than IETF-defined modules
* Follow a linear version history without branching
* Have simpler structure than general-purpose YANG modules, which simplifies classification of changes

Because IANA-maintained YANG modules are always expected to follow a linear version history without branching, the __COMPAT_ modifier defined in [I-D.ietf-netmod-yang-semver] is not needed or used for these modules. The __COMPAT_ modifier is only required for non-linear branched histories of YANG module versions. Therefore, only the _MAJOR.MINOR.PATCH_ elements of YANG Semver need be considered for IANA-maintained modules.

6.3. Rules Applicable to IANA-Maintained Modules

IANA-maintained YANG modules typically contain only enumerations (enum) and identity definitions, as they represent simple registry mappings. The most relevant compatibility rules for these modules are:

Backwards-Compatible Changes:

* Adding a new enum value or identity
* Changing status from "current" to "deprecated"
* Adding or updating "reference" statements
* Clarifying "description" statements without changing meaning
* Removing schema nodes that have status "obsolete" (per Section 3.1.1 of [I-D.ietf-netmod-yang-module-versioning])

Non-Backwards-Compatible Changes:

* Removing an enum value or identity (unless status is "obsolete")
* Changing status to "obsolete"
* Renaming an enum or identity
* Changing the numeric value assigned to an enum
* Reusing a previously assigned numeric value for a different enum
* Modifying "description" in a way that changes the semantic meaning

Editorial Changes:

* Fixing typographical errors in description text
* Updating contact information

* Formatting improvements

6.4. Process for Updating IANA-Maintained Modules

When a change is made to an IANA registry that has a corresponding YANG module, IANA MUST update the module following these steps:

6.4.1. Step 1: Follow RFC-Defined Rules

First, consult the RFC that defines the IANA registry and its associated YANG module. That RFC may specify:

* Specific rules for how registry entries map to YANG constructs
* Guidance on when and how to update the module
* Contact information for expert reviewers
* Special considerations for the particular registry

Always follow the specific guidance in the RFC that created the registry and module.

6.4.2. Step 2: Identify the Registry Change

Determine exactly what changed in the registry:

* Was a new entry added?
* Was an existing entry modified (description, reference, status)?
* Was an entry deprecated or obsoleted?
* Was an entry removed?
* Were multiple changes made simultaneously?

6.4.3. Step 3: Apply Changes to the YANG Module

Update the YANG module to reflect the registry changes. For IANA-maintained modules, this typically involves:

* Adding a new enum value or identity for new registry entries
* Updating description or reference statements for modified entries
* Changing status statements for deprecated or obsoleted entries

* Removing entries only if they are obsolete or if the defining RFC specifies removal

6.4.4. Step 4: Use Tooling to Determine the Version

Use the tools described in Appendix A to compare the updated module with the previously published version. The pyang --check-update-from command will identify any non-backwards-compatible changes:

```
pyang --check-update-from iana-if-type@2025-10-15.yang \
                          iana-if-type@2025-11-15.yang
```

If the tool reports NBC violations, the MAJOR version must be incremented and the rev:non-backwards-compatible extension must be added.

If the tool reports no violations, determine whether the change is backwards-compatible (BC) or editorial:

* *Editorial*: Only documentation changed (descriptions, references) without semantic meaning change → PATCH increment

* *BC*: New functionality added (new enums, identities) or status changed to deprecated → MINOR increment

6.4.5. Step 5: Update the Revision Statement

Add a new revision statement at the top of the module with:

* The current date

* The new version number (calculated based on the change classification)

* A clear description of what changed

* The rev:non-backwards-compatible extension if NBC changes occurred

6.4.6. Step 6: Validate the Module

Use validation tools to ensure the updated module is syntactically correct:

```
pyang --ietf iana-if-type@2025-11-15.yang
```

or

```
yanglint iana-if-type@2025-11-15.yang
```

6.4.7. Step 7: Seek Expert Review if Needed

In most cases, the classification will be straightforward. However, if any of the following apply, IANA SHOULD seek expert guidance as described in Section 7:

* The change classification is unclear
* Multiple simultaneous changes have different classifications
* Description changes may alter semantic meaning
* The tool output is unexpected or contradictory
* The situation is not covered by examples in Appendix B

6.4.8. Step 8: Publish the Updated Module

Once the module is validated and the version is confirmed:

* Publish the updated module to the IANA website
* Update any relevant registries or indexes
* Ensure the new version is discoverable and accessible

6.5. Simplified Decision Process

For IANA-maintained modules, the following simplified decision process can be used in most cases:

Did you add a new enum or identity? → Backwards-Compatible: MINOR version increment (e.g., 1.0.0 → 1.1.0)

Did you only update references or clarify descriptions without changing meaning? → Editorial: PATCH version increment (e.g., 1.0.0 → 1.0.1)

Did you change status from "current" to "deprecated"? → Backwards-Compatible: MINOR version increment (e.g., 1.0.0 → 1.1.0)

Did you change status to "obsolete", remove an entry, rename an entry, or change a value number? → Non-Backwards-Compatible: MAJOR version increment (e.g., 1.0.0 → 2.0.0) and ADD rev:non-backwards-compatible extension

Did you change a description in a way that changes behavior or meaning? → Non-Backwards-Compatible: MAJOR version increment (e.g., 1.0.0 → 2.0.0) and ADD rev:non-backwards-compatible extension

Are you uncertain? → Use tooling (Appendix A), consult scenarios (Appendix B), or seek expert guidance (Section 7)

6.6. Examples

For detailed examples of common scenarios (adding entries, updating references, deprecating entries, etc.), see Appendix B.

7. Seeking Expert Guidance

7.1. When to Seek Guidance

The RFC Editor and IANA SHOULD contact YANG experts in the following situations:

1. *Classification Uncertainty* - When it’s unclear whether a change is NBC, BC, or Editorial
2. *Complex Changes* - Multiple simultaneous changes that are difficult to classify individually
3. *Description Changes* - When a description update may alter the semantic meaning
4. *Unusual Situations* - Any scenario not clearly covered in this document
5. *Registry Restructuring* - Major changes to how a registry is organized
6. *Value Reuse* - When considering reusing a previously assigned value number
7. *Tool Disagreement* - When validation tools give unexpected or contradictory results
8. *RFC Editor Processing* - When RFC Editor changes may go beyond editorial scope

7.2. How to Seek Guidance

7.2.1. Primary Contact: YANG Doctors

Email the YANG Doctors mailing list:

* *Email*: yang-doctors@ietf.org
* *Purpose*: Technical review and guidance on YANG modules
* *Response Time*: Typically 1-2 weeks

When emailing, please include: - Description of the change or situation - The affected YANG module and relevant excerpts - Your proposed classification and version change - Specific questions or concerns - Any relevant tool output

7.2.2. Secondary Contact: NETMOD WG

For broader issues or when needing working group discussion: - *Email*: netmod@ietf.org - *Purpose*: Working group discussion of YANG issues

* *Primary Recipient*: Management Area Director (currently responsible for NETMOD)
* *Copy*: Operations Area Director
* *Purpose*: Escalation and prioritization when expert guidance is not received
* *When to Escalate*: After 2-3 weeks without response from YANG Doctors, or immediately for time-sensitive issues blocking RFC publication or critical IANA registry updates

7.3. Example Request

Subject: YANG Versioning Question - iana-if-type Update

Dear YANG Doctors,

I need guidance on classifying a change to the iana-if-type module.

Change Description:
The Interface Types registry has updated the description for interface type 6 (ethernet) to clarify that it includes both 10BASE-T and 100BASE-T variants.

Proposed YANG Change:
Update the description statement for the "ethernet" enum to include the clarification.

Question:
Should this be classified as Editorial (PATCH increment) since it’s clarifying existing behavior, or as BC (MINOR increment) because it’s adding new information?

The old description said: "Ethernet interface"
The new description says: "Ethernet interface, including 10BASE-T and 100BASE-T variants"

Current module version: 1.5.0
Proposed version: 1.5.1 (if Editorial) or 1.6.0 (if BC)

Thank you for your guidance.

8. Operational Considerations

This entire document provides operational guidance for the RFC Editor and IANA on how to process and publish YANG modules. The procedures described in Section 5 and Section 6 are designed to ensure consistent and correct versioning of YANG modules across all IETF and IANA publications.

Correct versioning is critical because consumers of YANG modules rely on the semantic version number to understand the compatibility and risk associated with updating to a new module version:

* *PATCH version increments* signal that only editorial changes have been made, indicating very low risk for updates

* *MINOR version increments* signal backwards-compatible additions, indicating that existing implementations will continue to work but new features are available

* *MAJOR version increments* signal non-backwards-compatible changes, indicating that implementations must carefully evaluate the impact before updating

Following the guidance in this document helps ensure that version numbers accurately communicate these compatibility expectations to the YANG module consumer community.

When uncertain about the correct classification or version for a module, the operational recommendation is to choose the more conservative option:

* If uncertain between editorial and backwards-compatible, choose backwards-compatible (MINOR rather than PATCH)

* If uncertain between backwards-compatible and non-backwards-compatible, choose non-backwards-compatible (MAJOR rather than MINOR, and include the NBC extension)

This conservative approach ensures that consumers are appropriately warned about potential compatibility implications, even if the actual risk turns out to be lower than indicated.

9. Security Considerations

This document gives instructions to IANA on how to handle YANG modules that are published in RFCs and also YANG modules that are derived from IANA registries.

Incorrect interpretion of this document could cause incorrect handling or versioning of IANA maintained YANG modules.

This document recommends the usage of various tools. Bugs or attacks on these tools could cause the tools to give incorrect or misleading guidance. In all cases, secondary evaluation of output of the tools should be performed to confirm that they are giving the anticipated results. The _YANG Doctors_ team can also be contacted for further advice, if required.

10. IANA Considerations

This document provides operational guidance to IANA and the RFC Editor for managing YANG modules. It does not require IANA to create or modify any registries, nor does it define any new registration procedures.

The guidance in this document is intended to clarify and standardize how IANA processes YANG modules in the "YANG Module Names" registry (https://www.iana.org/assignments/yang-parameters/ (https://www.iana.org/assignments/yang-parameters/)) and how IANA maintains YANG modules derived from IANA registries.

IANA should follow the procedures described in Section 5 when processing YANG modules from RFCs and the procedures described in Section 6 when updating IANA-maintained YANG modules.

11. References

11.1. Normative References

[I-D.ietf-netmod-rfc8407bis]
Bierman, A., Boucadair, M., and Q. Wu, "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", Work in Progress, Internet-Draft, draft-ietf-netmod-rfc8407bis-28, 5 June 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-rfc8407bis-28>.

[I-D.ietf-netmod-yang-module-filename]
Andersson, P., "YANG module file name convention", Work in Progress, Internet-Draft, draft-ietf-netmod-yang-module-filename-03, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-yang-module-filename-03>.

[I-D.ietf-netmod-yang-module-versioning]
Wilton, R., Rahman, R., Lengyel, B., Clarke, J., and J. Sterne, "Updated YANG Module Revision Handling", Work in Progress, Internet-Draft, draft-ietf-netmod-yang-module-versioning-15, 18 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-yang-module-versioning-15>.

[I-D.ietf-netmod-yang-semver]
Clarke, J., Wilton, R., Rahman, R., Lengyel, B., Sterne, J., and B. Claise, "YANG Semantic Versioning", Work in Progress, Internet-Draft, draft-ietf-netmod-yang-semver-24, 29 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-yang-semver-24>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/rfc/rfc7950>.

11.2. Informative References

[I-D.boucadair-veloce-yang]
Boucadair, M., "YANG deVELpment PrOCEss & maintenance (VELOCE)", Work in Progress, Internet-Draft, draft-boucadair-veloce-yang-05, 18 September 2025, <https://datatracker.ietf.org/doc/html/draft-boucadair-veloce-yang-05>.

[I-D.ietf-netmod-yang-schema-comparison]
Andersson, P., Wilton, R., and M. Vako, "YANG Schema Comparison", Work in Progress, Internet-Draft, draft-ietf-netmod-yang-schema-comparison-05, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-yang-schema-comparison-05>.

Appendix A. Available Tooling

This appendix describes tooling available to assist the RFC Editor and IANA in validating and versioning YANG modules. The tools and capabilities described here reflect the state of tooling as of the publication of this document. Tool capabilities are expected to evolve over time, and newer or improved tools may become available. The NETMOD working group and YANG Doctors can provide updated guidance on current best practices for tooling.

A.1. YANG Validation Tools

A.1.1. pyang

Purpose: pyang is a comprehensive YANG validator and converter tool that can validate syntax, check for backwards-compatible violations, and generate documentation.

Primary Use Cases:

* Validating YANG module syntax and compliance with IETF conventions

* Detecting non-backwards-compatible changes between module versions

* Generating tree diagrams for documentation

Installation: pyang is available via PyPI (pip install pyang) or from https://github.com/mbj4668/pyang (https://github.com/mbj4668/pyang)

Basic Syntax Validation:

pyang --ietf module-name.yang

This command validates the module syntax and checks compliance with IETF-specific conventions. Output will show any errors or warnings. A module SHOULD have no errors and SHOULD minimize warnings before publication.

Checking for NBC Changes:

pyang --check-update-from old-module.yang new-module.yang

This command compares two versions of a module and reports any violations of the backwards-compatible update rules defined in [I-D.ietf-netmod-yang-module-versioning].

Interpreting Output:

* If the command reports no errors: The changes are backwards-compatible (BC) or editorial

* If the command reports errors like "the enum ’X’ has been removed": The changes are non-backwards-compatible (NBC), requiring a MAJOR version increment and the NBC extension

* To distinguish BC from editorial changes, manually review the changes (editorial affects only documentation without semantic meaning changes)

Example Output:

old-module.yang:15: error: the enum ’deprecated-value’ has been removed

This indicates an NBC change has occurred.

A.1.2. yanglint

Purpose: yanglint is a YANG validator and data manipulation tool from the libyang project, useful for validating modules and instance data.

Primary Use Cases:

* Validating YANG module syntax

* Checking cross-module dependencies

* Validating instance data against YANG schemas

Installation: yanglint is part of libyang, available from https://github.com/CESNET/libyang (https://github.com/CESNET/libyang)

Basic Syntax Validation:

yanglint module-name.yang

This command validates the module syntax. No output indicates successful validation; errors will be displayed if found.

Validating with Dependencies:

yanglint -p /path/to/yang/modules module-name.yang

The -p option specifies a directory containing imported modules.

Interpreting Output:

* No output: Module is syntactically valid

* Error messages: Module has syntax errors that must be corrected before publication

A.1.3. YANG Catalog Tools

Purpose: The YANG Catalog (https://www.yangcatalog.org (https://www.yangcatalog.org)) provides online tools for module validation, comparison, and discovery.

Primary Use Cases:

* Validating YANG modules without local tool installation

* Comparing different versions of modules

* Viewing module dependencies and impact analysis

* Searching for existing modules and versions

Usage:

Access the web interface at https://www.yangcatalog.org (https://www.yangcatalog.org) and use the "Validator" and "YANG Impact Analysis" tools.

Interpreting Results:

The online tools provide visual feedback on validation results and module comparisons. The impact analysis tool can show which other modules depend on a given module, helping assess the impact of changes.

A.2. Recommended Workflow

The following workflow is recommended for validating and versioning YANG modules:

1. *Make Changes to Module* - Update the YANG file based on registry changes or RFC Editor edits

2. *Validate Syntax* - Run pyang or yanglint to check for syntax errors: ~~~~ shell pyang --ietf module-name.yang ~~~~

3. *Check for NBC Changes* - Use pyang to compare with the previous version: ~~~~ shell pyang --check-update-from old-version.yang new-version.yang ~~~~

4. *Review Tool Output* - Analyze any reported issues:

   * Errors from --check-update-from indicate NBC changes

   * No errors indicate BC or editorial changes

5. *Determine Version* - Based on the tool output and manual review:

   * NBC changes → MAJOR version increment (e.g., 1.0.0 → 2.0.0)

   * BC changes (new functionality) → MINOR version increment (e.g., 1.0.0 → 1.1.0)

   * Editorial changes (documentation only) → PATCH version increment (e.g., 1.0.0 → 1.0.1)

6. *Add Revision Statement* - Include:

   * Current date

   * New version number using ysv:version

   * Clear description of changes

   * rev:non-backwards-compatible extension if NBC changes occurred

7. *Final Validation* - Validate the complete updated module: ~~~~ shell pyang --ietf module-name.yang ~~~~

8. *Seek Review if Needed* - Contact experts (Section 7) if:

    * Tool output is unclear or surprising

    * Classification is uncertain

    * Description changes may have altered semantic meaning

A.3. Tool Limitations

While tools are valuable for YANG module validation and versioning, they have important limitations:

Limitation 1: Cannot Always Distinguish Editorial from BC Changes

Tools cannot determine whether a description change is purely editorial (clarifying existing meaning) or backwards-compatible (adding new information). Human judgment is required to make this distinction.

Example: Changing "Ethernet interface" to "Ethernet interface, including 10BASE-T and 100BASE-T" could be editorial (if those variants were always included) or BC (if documenting newly supported variants).

Limitation 2: Cannot Detect Semantic Changes in Descriptions

If a description change alters the intended behavior or meaning of a schema node, it may be an NBC change, but tools cannot detect this automatically.

Example: Changing "includes IPv4 only" to "includes IPv4 and IPv6" changes the semantic meaning and may be NBC, but tools will not flag this.

Limitation 3: Cannot Assess Practical Impact

Tools can identify that an NBC change has occurred but cannot assess how many implementations are affected or what the practical migration cost will be.

Limitation 4: May Produce False Positives or False Negatives

Tool implementations may have bugs or may not cover all edge cases in the YANG versioning rules. Always review tool output critically and consult experts if results are unexpected.

Recommendation: Always combine tool usage with the classification guidance in this document (particularly Appendix B) and seek expert review (Section 7) when uncertainty remains.

A.4. Future Tool Development

The NETMOD working group continues to develop improved tooling for YANG module management. Anticipated future capabilities include:

* Automated registry-to-YANG conversion tools
* Enhanced NBC change detection optimized for IANA-maintained modules
* Automated version recommendation based on detected changes
* Better detection of semantic changes in descriptions and constraints

As new tools become available, the NETMOD working group and YANG Doctors will provide guidance on their usage. The RFC Editor and IANA will be informed of tool updates that affect the workflows described in this document.

Appendix B. Summary of Registry Action Scenarios

This appendix provides a comprehensive reference of common scenarios encountered when updating YANG modules, particularly IANA-maintained modules. Each scenario describes the registry action, the corresponding YANG module change, the classification (NBC/BC/Editorial), the version change required, and whether the rev:non-backwards-compatible extension must be added.

B.1. Quick Reference Table

Registry Action	YANG Change	Classification	Version	NBC Ext
Add new registration	Add enum/ identity	BC	MINOR	No
Update reference (obsoleted RFC)	Update reference	Editorial	PATCH	No
Add additional reference	Update reference	Editorial	PATCH	No

Draft → RFC reference	Update reference	Editorial	PATCH	No
Deprecate (keep name)	status deprecated	BC	MINOR	No
Obsolete entry	status obsolete	NBC	MAJOR	Yes
Remove entry completely	Remove enum/ identity	NBC	MAJOR	Yes
Deprecate + remove name	Remove enum/ identity	NBC	MAJOR	Yes
Change value number	Change value	NBC	MAJOR	Yes
Reuse old value	Add with old value	NBC	MAJOR	Yes
Update description (clarify)	Update description	Editorial	PATCH	No
Update description (change meaning)	Update description	NBC	MAJOR	Yes
Rename entry	Change identifier	NBC	MAJOR	Yes
Add footnote	Optionally update	Editorial	PATCH	No
Non-YANG field changes	No change	N/A	None	No
Errata	Depends on content	Analyze	Varies	Maybe
Early alloc expired (left as-is)	No change	N/A	None	No

Early alloc expired (removed)	Follow removal rules	NBC	MAJOR	Yes
Revive expired allocation	Add enum/ identity	BC	MINOR	No

Table 1

Key: - *BC* = Backwards-Compatible; *NBC* = Non-Backwards-Compatible - *MAJOR/MINOR/PATCH* refer to the YANG Semver version components - *NBC Ext* = Whether rev:non-backwards-compatible extension is required - *Varies* or *Maybe* indicates the specific change must be analyzed using the detailed scenarios below

B.2. Detailed Scenarios

B.2.1. Scenario 1: Adding a New Registry Entry

Registry Action: A new entry is added to an IANA registry.

YANG Module Change: Add a new enum value or identity.

Classification: Backwards-Compatible (BC)

Version Change: Increment MINOR version (e.g., 1.0.0 → 1.1.0)

NBC Extension Required: No

Example:

```
   // Previous version 1.0.0
   typedef interface-type {
     type enumeration {
       enum ethernet {
         value 6;
         description "Ethernet interface";
       }
     }
   }

   // New version 1.1.0
   revision 2025-11-15 {
     ysv:version "1.1.0";
     description "Added wifi interface type";
   }

   typedef interface-type {
     type enumeration {
       enum ethernet {
         value 6;
         description "Ethernet interface";
       }
       enum wifi {
         value 71;
         description "IEEE 802.11 wireless interface";
       }
     }
   }
```

B.2.2. Scenario 2: Updating References

Registry Action: A reference is updated (e.g., RFC obsoleted, additional reference added, draft reference changed to RFC number).

YANG Module Change: Update the "reference" statement.

Classification: Editorial

Version Change: Increment PATCH version (e.g., 1.0.0 → 1.0.1)

NBC Extension Required: No

Rationale: Per [RFC7950] Section 11, reference statements may be added or updated without affecting compatibility.

Example:

```
// Previous version 1.0.0
enum foo {
  value 42;
  description "Foo interface type";
  reference "RFC 1234";
}

// New version 1.0.1
revision 2025-11-15 {
  ysv:version "1.0.1";
  description "Updated reference for RFC obsolescence";
}

enum foo {
  value 42;
  description "Foo interface type";
  reference "RFC 5678 (obsoletes RFC 1234)";
}
```

B.2.3. Scenario 3: Deprecating a Registry Entry

Registry Action: A registry entry is marked as deprecated (name and description remain visible).

YANG Module Change: Change status from "current" to "deprecated".

Classification: Backwards-Compatible (BC)

Version Change: Increment MINOR version (e.g., 1.0.0 → 1.1.0)

NBC Extension Required: No

Rationale: Changing status to deprecated is explicitly allowed as BC per Section 3.1.1.8 of [I-D.ietf-netmod-yang-module-versioning].

Example:

```
   // Previous version 1.0.0
   enum oldtype {
     value 99;
     status current;
     description "Old interface type";
   }

   // New version 1.1.0
   revision 2025-11-15 {
     ysv:version "1.1.0";
     description "Deprecated oldtype interface";
   }

   enum oldtype {
     value 99;
     status deprecated;
     description
       "Old interface type. This value is deprecated and
        SHOULD NOT be used in new implementations.";
   }
```

B.2.4. Scenario 4: Obsoleting a Registry Entry

Registry Action: A registry entry is marked as obsolete.

YANG Module Change: Change status from "deprecated" (or "current") to "obsolete".

Classification: Non-Backwards-Compatible (NBC)

Version Change: Increment MAJOR version (e.g., 1.0.0 → 2.0.0)

NBC Extension Required: Yes

Rationale: Changing status to obsolete indicates the value MUST NOT be used, breaking compatibility.

Example:

```
// Previous version 1.0.0
enum removedtype {
  value 98;
  status deprecated;
  description "Type being removed";
}

// New version 2.0.0
revision 2025-11-15 {
  ysv:version "2.0.0";
  rev:non-backwards-compatible;
  description
    "Obsoleted removedtype interface. Support has been removed.";
}

enum removedtype {
  value 98;
  status obsolete;
  description
    "Obsolete interface type. This value MUST NOT be used.
     Support was removed as of 2025-11-15.";
}
```

B.2.5. Scenario 5: Removing a Registry Entry Completely

Registry Action: A registry entry is removed with no trace.

YANG Module Change: Remove the enum or identity from the module.

Classification: Non-Backwards-Compatible (NBC)

Version Change: Increment MAJOR version (e.g., 1.0.0 → 2.0.0)

NBC Extension Required: Yes

Rationale: Removing a schema node is NBC per Section 3.1.2.1 of [I-D.ietf-netmod-yang-module-versioning].

Note: This is strongly discouraged. Prefer marking as "obsolete" instead.

B.2.6. Scenario 6: Renaming a Registry Entry

Registry Action: The name of a registry entry is changed.

YANG Module Change: Change the enum or identity identifier.

Classification: Non-Backwards-Compatible (NBC)

Version Change: Increment MAJOR version (e.g., 1.0.0 → 2.0.0)

NBC Extension Required: Yes

Rationale: Renaming breaks programmatic references to the identifier.

B.2.7. Scenario 7: Changing a Value Number

Registry Action: The numeric value assigned to a registry entry is changed.

YANG Module Change: Change the value assigned to an enum.

Classification: Non-Backwards-Compatible (NBC)

Version Change: Increment MAJOR version (e.g., 1.0.0 → 2.0.0)

NBC Extension Required: Yes

Rationale: Changing values breaks compatibility for implementations using those values.

Note: This is rare and strongly discouraged.

B.2.8. Scenario 8: Updating Description (Clarification)

Registry Action: Description is updated to clarify existing behavior without changing meaning.

YANG Module Change: Update the description statement.

Classification: Editorial

Version Change: Increment PATCH version (e.g., 1.0.0 → 1.0.1)

NBC Extension Required: No

Example: Changing "Ethernet interface" to "Ethernet interface, including 10BASE-T, 100BASE-T, and 1000BASE-T variants" where those variants were always included.

B.2.9. Scenario 9: Updating Description (Semantic Change)

Registry Action: Description is updated in a way that changes the semantic meaning or behavior.

YANG Module Change: Update the description statement.

Classification: Non-Backwards-Compatible (NBC)

Version Change: Increment MAJOR version (e.g., 1.0.0 → 2.0.0)

NBC Extension Required: Yes

Example: Changing "supports IPv4 only" to "supports IPv4 and IPv6" changes the expected behavior.

B.2.10. Scenario 10: Handling Errata

Registry Action: An errata report is filed for the registry or module.

YANG Module Change: Depends on the specific errata content.

Classification: Analyze the actual change, not the source (errata vs. new RFC does not determine classification).

Version Change: Follow the rules based on the actual change type.

NBC Extension Required: May be required depending on the change.

Examples: - Errata fixes typo in description → Editorial / PATCH - Errata adds missing enum → BC / MINOR - Errata corrects wrong value assignment → NBC / MAJOR

B.3. Notes on Classification

Important Principle: The source or trigger of a change (errata, new RFC, registry update, expert review, etc.) does NOT determine whether it is NBC, BC, or Editorial. What matters is the actual change made to the YANG module content.

Appendix C. Example IANA-Maintained Module

This appendix shows an example of a well-structured IANA-maintained YANG module, demonstrating proper use of versioning, revision statements, and the NBC extension.

C.1. Example: iana-if-type Module Structure

```
module iana-if-type {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:iana-if-type";
  prefix ianaift;

  import ietf-yang-revisions { prefix rev; }
```

```
     import ietf-yang-semver { prefix ysv; }

     organization
       "Internet Assigned Numbers Authority (IANA)";

     contact
       "Internet Assigned Numbers Authority

        ICANN
        12025 Waterfront Drive, Suite 300
        Los Angeles, CA 90094

        Tel: +1 424 254 5300

        <mailto:iana@iana.org>

        See the Interface Types (ifType) registry:
        <https://www.iana.org/assignments/smi-numbers>";

     description
       "This YANG module defines the iana-if-type typedef, which
        contains YANG definitions for IANA-registered interface types.

        This YANG module is maintained by IANA and reflects the
        'Interface Types (ifType)' registry.

        Copyright (c) 2025 IETF Trust and the persons identified as
        authors of the code.  All rights reserved.

        Redistribution and use in source and binary forms, with or
        without modification, is permitted pursuant to, and subject
        to the license terms contained in, the Revised BSD License
        set forth in Section 4.c of the IETF Trust's Legal Provisions
        Relating to IETF Documents
        (https://trustee.ietf.org/license-info).

        This version of this YANG module is part of RFC XXXX; see
        the RFC itself for full legal notices.";

     // Example revision with new interface type (BC change)
     revision 2025-11-15 {
       ysv:version "2.5.0";
       description
         "Added new interface type 'wifi6e' per registry update.";
       reference
         "RFC XXXX: IANA Guidance for YANG Modules";
     }
```

```
     // Example revision with obsoleted type (NBC change)
     revision 2025-10-01 {
       ysv:version "2.0.0";
       rev:non-backwards-compatible;
       description
         "Obsoleted 'arcnet' interface type as support has been removed.";
       reference
         "RFC XXXX: IANA Guidance for YANG Modules";
     }

     // Example earlier revision (BC change)
     revision 2025-09-01 {
       ysv:version "1.1.0";
       description
         "Added new interface type 'virtualEthernet'.";
     }

     // Example initial revision
     revision 2025-01-01 {
       ysv:version "1.0.0";
       description
         "Initial version matching the Interface Types registry
          as of 2025-01-01.";
     }

     typedef interface-type {
       type enumeration {
         enum other {
           value 1;
           description
             "None of the following types.";
         }
         enum ethernet {
           value 6;
           description
             "Ethernet interface, including 10BASE-T, 100BASE-T,
              and 1000BASE-T variants.";
           reference
             "RFC 3635: Definitions of Managed Objects for the
              Ethernet-like Interface Types";
         }
         enum wifi6e {
           value 289;
           description
             "IEEE 802.11ax (Wi-Fi 6E) wireless interface.";
           reference
             "IEEE 802.11ax-2021";
         }
```

```
        enum arcnet {
          value 35;
          status obsolete;
          description
            "ARCNET interface. This interface type is obsolete
             and MUST NOT be used. Support was removed 2025-10-01.";
        }
      }
      description
        "This typedef is used to represent the IANA-registered
         interface types. It is derived from the 'Interface Types
         (ifType)' registry.";
      reference
        "IANA Interface Types registry:
         <https://www.iana.org/assignments/smi-numbers>";
    }
  }
```

Acknowledgments

The creation of this document was motivated by questions from IANA team members Amanda Baber and Sabrina Tanamal regarding the correct handling of YANG module versioning. This was followed by a productive in-person discussion at IETF 124 between members of the YANG versioning design team, the IANA team, NETMOD working group chairs, RFC Editor representatives, and the OPS Area Director.

Participants in that meeting included: Amanda Baber, Jason Sterne, Joe Clarke, Kent Watsen, Lou Berger, Mahesh Jethanandani, Per Andersson, Reshad Rahman, Rob Wilton, and Sabrina Tanamal.

Special thanks to Joe Clarke for his presentation on YANG versioning tooling at IETF 124, which informed the tooling guidance in Appendix A.

The authors thank the RFC Editor and IANA teams for their collaboration in refining the operational procedures described in this document.

The authors also thank the NETMOD working group for their extensive work on YANG versioning specifications that form the foundation of this guidance, including the module versioning framework, semantic versioning, and associated tooling.

The initial substantive revision of this document used Claude Sonnet 4.5 to create prose and examples, which have been subsequently reviewed and refined by the YANG Versioning design team and the NETMOD working group.

Author’s Address

Robert Wilton
Cisco
Email: rwilton@cisco.com