

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: June 23, 2019

R. Wilton  
Cisco Systems, Inc.  
December 20, 2018

YANG Packages  
draft-rwilton-netmod-yang-packages-00

## Abstract

This document defines YANG packages, an organizational structure holding a set of related YANG modules, that can be used to simplify the conformance and sharing of YANG schema. It describes how YANG instance data documents are used to define YANG packages, and how the YANG library information published by a server can be augmented with additional packaging related information.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 23, 2019.

## Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of

Internet-Draft

YANG Packages

December 2018

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	<a href="#">Terminology and Conventions . . . . .</a>	<a href="#">2</a>
<a href="#">2.</a>	<a href="#">Introduction . . . . .</a>	<a href="#">3</a>
<a href="#">3.</a>	<a href="#">Background on YANG packaging . . . . .</a>	<a href="#">4</a>
<a href="#">4.</a>	<a href="#">Objectives . . . . .</a>	<a href="#">4</a>
<a href="#">5.</a>	<a href="#">Package description . . . . .</a>	<a href="#">5</a>
<a href="#">5.1.</a>	<a href="#">Package definition rules . . . . .</a>	<a href="#">6</a>
<a href="#">5.2.</a>	<a href="#">Package versioning . . . . .</a>	<a href="#">7</a>
<a href="#">5.3.</a>	<a href="#">Client server package conformance . . . . .</a>	<a href="#">8</a>
<a href="#">5.4.</a>	<a href="#">Submodules packaging considerations . . . . .</a>	<a href="#">8</a>
<a href="#">5.5.</a>	<a href="#">Revision history . . . . .</a>	<a href="#">9</a>
<a href="#">5.6.</a>	<a href="#">Uniqueness of packages and global registry . . . . .</a>	<a href="#">9</a>
<a href="#">6.</a>	<a href="#">YANG Packaging instance data . . . . .</a>	<a href="#">9</a>
<a href="#">7.</a>	<a href="#">YANG Packaging additions to YANG library . . . . .</a>	<a href="#">11</a>
<a href="#">7.1.</a>	<a href="#">Package List . . . . .</a>	<a href="#">11</a>
<a href="#">7.2.</a>	<a href="#">Binding from schema to package . . . . .</a>	<a href="#">11</a>
<a href="#">7.3.</a>	<a href="#">Tree diagram . . . . .</a>	<a href="#">12</a>
<a href="#">8.</a>	<a href="#">YANG Packaging groupings . . . . .</a>	<a href="#">12</a>
<a href="#">9.</a>	<a href="#">YANG Modules . . . . .</a>	<a href="#">14</a>
<a href="#">10.</a>	<a href="#">Open Questions/Issues . . . . .</a>	<a href="#">26</a>
<a href="#">11.</a>	<a href="#">Security Considerations . . . . .</a>	<a href="#">26</a>
<a href="#">12.</a>	<a href="#">IANA Considerations . . . . .</a>	<a href="#">27</a>
<a href="#">13.</a>	<a href="#">References . . . . .</a>	<a href="#">27</a>
<a href="#">13.1.</a>	<a href="#">Normative References . . . . .</a>	<a href="#">27</a>
<a href="#">13.2.</a>	<a href="#">Informative References . . . . .</a>	<a href="#">29</a>
<a href="#">Appendix A.</a>	<a href="#">Tree output for ietf-yang-library with package augmentations . . . . .</a>	<a href="#">29</a>
<a href="#">Appendix B.</a>	<a href="#">Examples . . . . .</a>	<a href="#">31</a>
<a href="#">B.1.</a>	<a href="#">Example IETF Network Device YANG package . . . . .</a>	<a href="#">31</a>
<a href="#">B.2.</a>	<a href="#">Example IETF Basic Routing YANG package . . . . .</a>	<a href="#">34</a>
<a href="#">B.3.</a>	<a href="#">Package import conflict resolution example . . . . .</a>	<a href="#">37</a>
	<a href="#">Author's Address . . . . .</a>	<a href="#">39</a>

## [1.](#) Terminology and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] when, and only when, they appear in all

capitals, as shown here.

This document uses terminology introduced in the YANG versioning requirements draft [[I-D.verdt-netmod-yang-versioning-reqs](#)].

This document also makes of the following terminology introduced in the Network Management Datastore Architecture [[RFC8342](#)]:

- o datastore schema

In addition, this document makes use of the following terminology:

- o bc: Used as an abbreviation for a backwards-compatible change.
- o nbc: Used as an abbreviation for a non-backwards-compatible change.
- o editorial change: A backwards-compatible change that does not change the YANG module semantics in any way.

Note - the bc/nbc/editorial terminology should probably be defined and referenced from the YANG module versioning solution draft.

## [2.](#) Introduction

This document defines and describes the YANG [[RFC7950](#)] constructs that are used to define and use YANG packages.

A YANG package is an organizational structure that groups a set of related YANG modules together into a consistent versioned definition. YANG packages can themselves refer to and reuse other package definitions.

The draft consists of the following significant sections:

A background section that describes some of the prior work in this area, both within IETF and the wider industry.

An overview of the objectives for a YANG packaging solution, and also what work is out of scope for this document.

The definition of YANG packages, how package definitions are constructed, and how they are used.

How YANG instance data documents

[\[I-D.ietf-netmod-yang-instance-file-format\]](#) are used to define particular YANG package instances.

Augmentations to the YANG library [\[I-D.ietf-netconf-rfc7895bis\]](#) content published by servers to include YANG packaging related information.

YANG modules the provide the definitions for YANG packages.

Non-normative examples of YANG package instances are provided in the appendices.

### [3.](#) Background on YANG packaging

It has long been acknowledged within the IETF NETMOD community that network management using YANG requires a unit of organization and conformance that is broader in scope than individual YANG modules.

'The YANG Package Statement' [\[I-D.bierman-netmod-yang-package\]](#) proposed a YANG package mechanism based on new YANG language statements, where a YANG package is defined in a file similar to how YANG modules are defined, and would require enhancements to YANG compilers to understand the new statements used to define particular package instances. This document did not progress in the working group, although this may have been due to other higher priority concerns or resource constraints within the working group rather than due to consideration of the technical merits of the proposed approach.

OpenConfig [\[openconfigsemver\]](#) describes an approach to versioning 'bundle releases' based on git tags. I.e. a set of modules, at particular versions, can be marked with the same release tag to indicate that they are known to interoperate together.

The NETMOD WG in general, and the YANG versioning design team in particular, are exploring solutions to the YANG versioning requirements, [\[I-D.verdt-netmod-yang-versioning-reqs\]](#). Solutions to the versioning requirements can be split into several distinct areas.

One draft, TBD ([draft-verdt-netmod-yang-semver](#)), has a primary focus on YANG versioning scoped to individual modules. But an overall solution should also consider YANG versioning and conformance scoped to a server's datastore schema. YANG packages may help form part of the solution for versioning at the datastore schema level.

#### 4. Objectives

The main goals of YANG package definitions include, but are not restricted to:

- o To act as a simplified YANG conformance mechanism. Rather than conformance being performed against a set of individual YANG module revisions, conformance could also be more simply stated in terms of YANG packages, with a set modifications (e.g. additional modules, deviations, or features).
- o To allow YANG datastore schema to be specified in a more concise way rather than having to list all modules and revisions. YANG

package definitions can be defined in documents that can be referenced by a URI rather than requiring explicit lists of modules to be shared between client and server. Hence, a YANG package must contain sufficient information to allow a client or server to precisely construct the schema associated with the package.

- o To provide generic packaging related YANG grouping definitions for use in other YANG modules, as required.

Protocol mechanisms of how clients could negotiate which packages or package versions are be used for client server communications are outside the scope of this document. However, the design of the YANG library augmentations for YANG packages are intended to keep open the possibility of such extensions in future work.

Finally, the package definitions proposed by this document are intended to be relatively basic in their definition and the functionality that they support. As indsutry gains experience using YANG packages, the standard YANG mechanisms of updating, or augmenting, YANG modules could be used to extend the functionality supported by YANG packages.

## [5.](#) Package description

This document specifies an approach to defining YANG packages that is different to either of the approaches described in the background.

The approach defined here is for a YANG package definition structure to be defined using existing YANG language statements without requiring extensions or new YANG statements. By making use of this structure, particular YANG package instances can be defined as YANG instance data documents [[I-D.ietf-netmod-yang-instance-file-format](#)] with well defined names and locations.

The YANG semantic versioning scheme, described in [draft-verdt-netmod-yang-semver](#) (TBD), is used to version YANG packages using an equivalent scheme to how individual YANG modules version numbers are changed.

YANG library is augmented to allow servers to report the packages that they implement and to associate those packages back to particular datastore schema.

TODO - It would be helpful if the YANG instance data file format [[I-D.ietf-netmod-yang-instance-file-format](#)] could also reference a YANG packages to specify the schema associated with an instance data

document. This could either be defined in instance-file-format draft, or as a YANG augmentation as part of this draft.

Each version of a YANG package defines: a set of YANG modules that are implemented at particular versions or revisions; a set of YANG modules that are import-only with particular versions or revisions; and a set of mandatory module features that implementations of the package MUST implement or otherwise deviate.

### [5.1.](#) Package definition rules

The following rules define how packages are defined:

Every YANG package definition MUST be referentially complete.  
I.e. all import and include statements for all YANG modules

included in a package MUST resolve to a module specified in the package itself, or an imported package.

For a given package, each separate instance of the package MUST have a unique version number that follows the semantic versioning rules described in [Section 5.2](#).

A package MAY have a revision-date. Any package revision-dates MUST be unique for different package versions.

For each module implemented by a package, only a single revision/version MUST be implemented.

The version/revision of a module listed in the package module list supercedes any version/revision of the module listed in a imported package module list. This allows a package to resolve any conflicting implemented module versions/revisions in imported packages.

The replaces-revision leaf-list in the import-only-module list can be used to exclude duplicate revisions of import-only modules from imported packages. Otherwise, the import-only-modules for a package are the import-only-modules from all imported packages combined with any modules listed in the packages import-only-module list.

Modules referenced by a package SHOULD specify the version of the module, both in the package definition and within the module definition itself.

Modules referenced by a package MUST specify the revision date of the module, both in the package definition and within the module definition itself.

## [5.2](#). Package versioning

Every YANG package must specify a YANG semantic version field that defines the particular version of the package.

The rules for incrementing the YANG package version number are equivalent to the semantic versioning rules used to version individual YANG modules, defined in TBD ([draft-verdt-netmod-yang-](#)

[semver](#)).

The semantic versioning rules, as they apply to YANG packages, are defined using the following two step process:

The first step is to determine whether the change to the YANG package is classified as a major, minor, or editorial based on the content that has changed in the package relative to the previous version. Where available, the semantic version number of the referenced elements in the package (imported packages or modules) can be used to help determine what type of change is being made. The formal rules are:

If any of the referenced elements of the package (imported packages or modules) are changed in an nbc way, or if any imported package, module, or mandatory-feature is removed from the package definition, then the package has been updated in an nbc way.

If none of the referenced elements of the package (imported packages, modules) are removed or changed in a nbc way, but some referenced elements are changed in a bc way, or new referenced elements or mandatory-features added, then the package is deemed to be updated in a bc way.

If none of the referenced elements of the package (imported packages, modules) are added, removed, or changed in a nbc or bc way, but some referenced elements have editorial changes then the package is deemed to be updated in an editorial way.

The second step, after it has been determined what type of version change is being made to the YANG package, is for the YANG semantic versioning rules to be applied to update the YANG package semantic version number. The formal rules are:

If the package is being updated in a nbc way, then the package version "X.Y.Z[m|M]" SHOULD be updated to "X+1.0.0" unless that package version has already been defined with different content, in which case the package version "X.Y.Z+1M" MUST be used instead.

If the package is being updated in a bc way, then the package



version "X.Y.Z[m|M]" SHOULD be updated to "X.Y+1.0" unless that package version has already been defined with different content, in which case if the current package version is "X.Y.ZM" then it MUST be updated to "X.Y.Z+1M", or otherwise "X.Y.Z+1m".

If the package is being updated in an editorial way, then the package version "X.Y.Z[m|M]" MUST be updated to "X.Y.Z+1[m|M]", retaining the 'm|M' character if it is already present in the previous version."

Package YANG semantic version numbers beginning with 0, i.e "0.X.Y" are regarded as beta definitions and need not follow the nbc rules, and the minor version number can be incremented instead.

In all cases, the 3 number fields that comprise a YANG semantic version number associated with a YANG package MUST uniquely identify the contents of that YANG package.

### [5.3.](#) Client server package conformance

The YANG semantic versioning scheme used for YANG packages means that a client can determine the nature of changes between two package revisions.

This means that a client is not restricted to working only with servers that advertise exactly the same version of package in YANG library. Instead, reasonable clients should be able to interoperate with a server that supports a package version that is backwards compatible to what the client is designed for.

For example, a client coded to support 'foo' package at version 1.0.0 should interoperate with a server implementing 'foo' package at version 1.3.5, because the YANG semantic versioning rules require that package version 1.3.5 is backwards compatible to version 1.0.0.

This also has a relevance on servers that are capable of supporting version selection because they need not necessarily support every version of a YANG package to ensure good client compatibility. Choosing suitable minor versions within each major version number should generally be sufficient, particular if they can avoid NBC patch level changes (i.e. 'M' labelled versions).

### [5.4.](#) Submodules packaging considerations

As defined in [[RFC7950](#)] and [draft-verdt-netmod-yang-semver](#) (TBD), YANG conformance and versioning is specified in terms of particular revisions of YANG modules rather than for individual submodules.

However, YANG package definitions also include the list of submodules included by a module, primarily to provide a location of where the submodule definition can be obtained from, allowing a YANG schema to be fully constructed from a YANG package instance-data definition.

Restructuring how a module uses, or does not use, submodules is treated as an editorial level change in YANG semantic versioning, on the condition that there is no change in the modules semantic behavior due to the restructuring.

To ensure that a module and any constituent submodule are tightly related, all 'include' statements in a YANG module SHOULD specify revision-dates of the included submodules. If 'include' statement revision-dates are included in the YANG module then they MUST match the 'revision' field specified for the submodule in the packages's submodules lists.

### [5.5.](#) Revision history

TODO - Probably eventually delete this section ...

YANG packages do not contain a revision history. It is anticipated that YANG packages versions may become branched over time and hence maintaining a linear revision history would likely be problematic and less useful. Further, if YANG packages versions are managed in a source control system, then additional version meta-data information could be stored in the source control system, which are generally capable of representing a branched revision history.

### [5.6.](#) Uniqueness of packages and global registry

The name given to a package SHOULD be globally unique, and it SHOULD include an appropriate organization prefix in the name, equivalent to YANG module naming conventions.

Each package MUST define a unique namespace. It is anticipated that a registry of package namespaces would be managed by IANA. It is unclear whether specific standard package versions would need to be managed in a similar way.

Ideally a YANG instance data document defining a particular package version would be publically available at one or more URIs.

## [6.](#) YANG Packaging instance data

YANG packages are expected to be defined as YANG instance data

documents [[I-D.ietf-netmod-yang-instance-file-format](#)] using the YANG schema below to define the package data itself.

The instance data document for each version of a YANG package SHOULD be made available at one of more locations accessible via a URI. If one of the listed locations defines a definitive reference implementation for the package definition then it MUST be listed as the first entry in the list.

The "ietf-yang-package" YANG module has the following structure:

```
module: ietf-yang-package
  +--ro yang-package
    +--ro name                yang:yang-identifier
    +--ro version              yang-sem-ver
    +--ro revision-date?      yanglib:revision-identifier
    +--ro location*            inet:uri
    +--ro description?         string
    +--ro reference?           string
    +--ro tag*                  tags:tag
    +--ro mandatory-feature*   string
    +--ro imported-packages* [name version]
      | +--ro name            yang:yang-identifier
      | +--ro version         yang-sem-ver
      | +--ro location*       inet:uri
    +--ro module* [name]
      | +--ro name            yang:yang-identifier
      | +--ro revision?       revision-identifier
      | +--ro version?        yang-sem-ver
      | +--ro namespace       inet:uri
      | +--ro location*       inet:uri
      | +--ro submodule* [name]
      |   +--ro name          yang:yang-identifier
      |   +--ro revision       yanglib:revision-identifier
      |   +--ro location*     inet:uri
    +--ro import-only-module* [name revision]
      +--ro name                yang:yang-identifier
      +--ro revision             union
      +--ro version?             yang-sem-ver
      +--ro namespace            inet:uri
      +--ro location*            inet:uri
```

```

+--ro submodule* [name]
|   +--ro name      yang:yang-identifier
|   +--ro revision   yanglib:revision-identifier
|   +--ro location*  inet:uri
+--ro replaces-revision*  yanglib:revision-identifier

```

## [7.](#) YANG Packaging additions to YANG library

### [7.1.](#) Package List

The main addition is a top level 'yang-library/package' list that lists all package of all versions known to the server. Each package itself is defined using imported packages and module-sets to define the specific set of modules implemented and imported by the package. The use of module-sets allows the module definitions to be shared with the existing YANG library schema definitions. The existing rule of RFC 7995bis related to combining modules-sets also applies here, i.e. The combined set of modules defined by the module-sets MUST NOT contain modules implemented at different revisions. I.e. the module-sets leaf-list is directly equivalent to the explicit module and import-only-module lists in the instance data YANG package definition.

The 'yang-library/package' list MAY include multiple versions of a particular package. E.g. if the server is capable of allowing clients to select which package versions should be used by the server.

### [7.2.](#) Binding from schema to package

The second augmentation is to allow a server to optionally indicate that a schema definition directly relates to a package. Since YANG packages are available offline, it may be sufficient for a client to only check that a compatible version of the YANG package is being implemented by the server without fetching and comparing the full module list.

If a server indicates that its schema maps to a particular package

then it MUST support all mandatory-features defined as part of that package, and it MUST NOT have any deviations to the modules defined by the package. A server MAY implement features not specified in the package's mandatory-features list.

If a server cannot faithfully implement a package then it can define a new package to accurately report what it does implement. The new package can include the original package as an imported package, and the new package can define additional modules containing deviations to the original package, allowing the new package to accurately describe the server behavior. There is no specific mechanism provided to indicate that a mandatory-feature is not supported on a server, but deviations MAY be used to disable functionality predicated by a mandatory-feature.

### [7.3.](#) Tree diagram

The "ietf-yang-library-packages" YANG module has the following structure:

```
module: ietf-yang-library-packages
  augment /yanglib:yang-library:
    +--ro package* [name version]
      +--ro name          yang:yang-identifier
      +--ro version       yang-sem-ver
      +--ro revision-date? yanglib:revision-identifier
      +--ro location*     inet:uri
      +--ro description?  string
      +--ro reference?    string
      +--ro tag*          tags:tag
      +--ro mandatory-feature* string
      +--ro imported-packages* [name version]
        | +--ro name      yang:yang-identifier
        | +--ro version   yang-sem-ver
      +--ro module-set*
        -> /yanglib:yang-library/module-set/name
  augment /yanglib:yang-library/yanglib:schema:
    +--ro package
      +--ro name?      -> /yanglib:yang-library/package/name
```

```

    +--ro version?    leafref
augment /yanglib:yang-library/yanglib:module-set/
                                yanglib:import-only-module:
    +--ro replaces-revision* yanglib:revision-identifier

```

## 8. YANG Packaging groupings

Groupings for YANG packaging related constructs are provided in a 'types' module for use by the instance-data and YANG library constructs described previously. They are also available to be used by other modules that have a need for packaging information.

The "ietf-yang-package-types" YANG module has the following structure:

```

module: ietf-yang-package-types

  grouping yang-pkg-identification-leafs
    +---- name          yang:yang-identifier
    +---- version       yang-sem-ver
  grouping yang-pkg-common-leafs

```

```

+---- revision-date?    yanglib:revision-identifier
+---- location*         inet:uri
+---- description?     string
+---- reference?       string
+---- tag*             tags:tag
+---- mandatory-feature* string
+---- imported-packages* [name version]
    +---- name          yang:yang-identifier
    +---- version       yang-sem-ver
grouping yang-pkg-library-definition
+---- name              yang:yang-identifier
+---- version           yang-sem-ver
+---- revision-date?    yanglib:revision-identifier
+---- location*         inet:uri
+---- description?     string
+---- reference?       string
+---- tag*             tags:tag
+---- mandatory-feature* string

```

```

+---- imported-packages* [name version]
|   +---- name          yang:yang-identifier
|   +---- version       yang-sem-ver
+---- module-set*
      -> /yanglib:yang-library/module-set/name
grouping yang-pkg-file-definition
+---- name                yang:yang-identifier
+---- version              yang-sem-ver
+---- revision-date?      yanglib:revision-identifier
+---- location*           inet:uri
+---- description?        string
+---- reference?          string
+---- tag*                tags:tag
+---- mandatory-feature*  string
+---- imported-packages* [name version]
|   +---- name          yang:yang-identifier
|   +---- version       yang-sem-ver
|   +---- location*     inet:uri
+---- module* [name]
|   +---- name          yang:yang-identifier
|   +---- revision?     revision-identifier
|   +---- version?      yang-sem-ver
|   +---- namespace     inet:uri
|   +---- location*     inet:uri
|   +---- submodule* [name]
|       +---- name?      yang:yang-identifier
|       +---- revision   yanglib:revision-identifier
|       +---- location*  inet:uri
+---- import-only-module* [name revision]
      +---- name?        yang:yang-identifier

```

```

+---- revision?          union
+---- version?           yang-sem-ver
+---- namespace          inet:uri
+---- location*          inet:uri
+---- submodule* [name]
|   +---- name?          yang:yang-identifier
|   +---- revision       yanglib:revision-identifier
|   +---- location*      inet:uri
+---- replaces-revision* yanglib:revision-identifier

```

## 9. YANG Modules

The YANG module definitions for the modules described in the previous sections.

```
<CODE BEGINS> file "ietf-yang-package-types@2018-11-26.yang"
module ietf-yang-package-types {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yang-package-types";
  prefix "pkg-types";

  import ietf-yang-types {
    prefix yang;
    reference "RFC 6991: Common YANG Data Types.";
  }
  import ietf-inet-types {
    prefix inet;
    reference "RFC 6991: Common YANG Data Types.";
  }
  import ietf-yang-library {
    prefix yanglib;
    reference "RFC 7895bis: YANG Library";
  }
  import ietf-module-tags {
    prefix tags;
    reference
      "XXX, (draft-ietf-netmod-module-tags-03): YANG Module Tags";
  }

  organization
    "IETF NETMOD (Network Modeling) Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/netmod/>
    WG List:  <mailto:netmod@ietf.org>
```



"This module provides type and grouping definitions for YANG packages.

Copyright (c) 2018 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 Simplified BSD License set forth in [Section 4.c](#) of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

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

```
// RFC Ed.: update the date below with the date of RFC publication
// and remove this note.
// RFC Ed.: replace XXXX with actual RFC number and remove this
// note.
revision 2018-11-26 {
    description
        "Initial revision";
    reference
        "RFC XXXX: YANG Schema Versioning.";
}

/*
 * Typedefs
 */

typedef yang-sem-ver {
    type string {
        pattern '\d+[\.]\d+[\.]\d+[mM]?';
    }
    description
        "Represents a YANG semantic version number.";
    reference
        "TODO - Should be defined by YANG versioning types module";
}

/*
 * Groupings
 */
```

```
grouping yang-pkg-identification-leafs {
  description
    "Parameters for identifying a specific version of a YANG
    package";

  leaf name {
    type yang:yang-identifier;
    mandatory true;
    description
      "The YANG package name.";
  }

  leaf version {
    type yang-sem-ver;
    mandatory true;
    description
      "YANG package version. Follows YANG semantic versions rules
      defined in XXX";
  }
}

grouping yang-pkg-common-leafs {
  description
    "Defines definitions common to all YANG package definitions.";

  leaf revision-date {
    type yanglib:revision-identifier;

    description
      "An optional revision identifier of when this package version
      was created. This does not need to be unique across all
      versions of a package.";
  }

  leaf-list location {
    type inet:uri;
    description
      "Contains a URL that represents where an instance data file
      for this YANG package can be found.

      This leaf will only be present if there is a URL
      available for retrieval of the schema for this entry.

      If multiple locations are provided, then the first location
      in the leaf-list MUST be the definitive location that
      uniquely identifies this package";
  }
}
```

```
leaf description {
    type string;

    description "Provides a description of the package";
}

leaf reference {
    type string;

    description "Allows for a reference for the package";
}

leaf-list tag {
    type tags:tag;
    description
        "Tags associated with a YANG package.  Module tags defined in
        XXX, ietf-netmod-module-tags can be used here but with the
        modification that the tag applies to the entire package
        rather than a specific module.  See the IANA 'YANG Module Tag
        Prefix' registry for reserved prefixes and the IANA 'YANG
        Module IETF Tag' registry for IETF standard tags.";
}

leaf-list mandatory-feature {
    type string;
    // TODO - Is there a better type for this?
    description
        "List all features from modules included in the package that
        MUST be supported by any server implementing the package.
        All other features defined in included packages are OPTIONAL
        to implement.

        Features are identified using <module-name>:<feature>";
}

list imported-packages {
    key "name version";
    description
        "An entry in this list represents a package that is imported
        as part of the package definition."
```

If packages implement different revisions or versions of the same module, then an explicit entry in the module list **MUST** be provided to select the specific module version 'implemented' by this package definition.

For import-only modules, the replaces-revision leaf-list can be used to select the specific module versions imported by

```
        this package.";
    reference
        "XXX";

    uses yang-pkg-identification-leafs;
}

grouping yang-pkg-file-definition {
    description
        "The set of parameters that describe a particular YANG package.";

    uses yang-pkg-identification-leafs;

    uses yang-pkg-common-leafs {
        augment "imported-packages" {
            description "Add the package location path";

            leaf-list location {
                type inet:uri;
                description
                    "Contains a URL that represents where an instance data
                     file for this YANG package can be found.

                    This leaf will only be present if there is a URL
                    available for retrieval of the schema for this entry.

                    If multiple locations are provided, then the first
                    location in the leaf-list MUST be the definitive location
                    that uniquely identifies this package";
            }
        }
    }
}
```

```

list module {
  key "name";
  description
    "An entry in this list represents a module that must be
    implemented by a server implementing this package, as per
    RFC 7950 section 5.6.5, with a particular set of supported
    features and deviations.

    A entry in this list overrides any module version
    'implemented' by an imported package";
  reference
    "RFC 7950: The YANG 1.1 Data Modeling Language.";

  uses yanglib:module-identification-leafs;

```

```

leaf version {
  type yang-sem-ver;
  description
    "The YANG module or submodule version. If no version
    statement is present in the YANG module or submodule, this
    leaf is not instantiated.";
}

leaf namespace {
  type inet:uri;
  mandatory true;
  description
    "The XML namespace identifier for this module.";
}
uses yanglib:location-leaf-list;

list submodule {
  key "name";
  description
    "Each entry represents one submodule within the
    parent module.";

  leaf name {
    type yang:yang-identifier;
    description
      "The YANG submodule name.";
  }
}

```

```

    }
    leaf revision {
        type yanglib:revision-identifier;
        mandatory true;
        description
            "The YANG submodule revision date.  If the parent module
            include statement for this submodule includes a revision
            date then it MUST match this leaf's value.";
    }

    uses yanglib:location-leaf-list;
}

list import-only-module {
    key "name revision";
    description
        "An entry in this list indicates that the server imports
        reusable definitions from the specified revision of the
        module, but does not implement any protocol accessible
        objects from this revision."
}

```

Multiple entries for the same module name MAY exist. This can occur if multiple modules import the same module, but specify different revision-dates in the import statements.";

```

leaf name {
    type yang:yang-identifier;
    description
        "The YANG module name.";
}
leaf revision {
    type union {
        type yanglib:revision-identifier;
        type string {
            length 0;
        }
    }
    description
        "The YANG module revision date.  A zero-length string is
        used if no revision statement is present in the YANG

```

```

        module.";
    }
    leaf version {
        type yang-sem-ver;
        description
            "The YANG module or submodule version.  If no version
            statement is present in the YANG module or submodule, this
            leaf is not instantiated.";
    }
    leaf namespace {
        type inet:uri;
        mandatory true;
        description
            "The XML namespace identifier for this module.";
    }

    uses yanglib:location-leaf-list;

    list submodule {
        key "name";
        description
            "Each entry represents one submodule within the
            parent module.";

        leaf name {
            type yang:yang-identifier;
            description
                "The YANG submodule name.";
        }
    }

```

```

    leaf revision {
        type yanglib:revision-identifier;
        mandatory true;
        description
            "The YANG submodule revision date.  If the parent module
            include statement for this submodule includes a revision
            date then it MUST match this leaf's value.";
    }

    uses yanglib:location-leaf-list;
}

```

```

    leaf-list replaces-revision {
      type yanglib:revision-identifier;
      description
        "Gives the revision of an import-only-module defined in
        an imported package that is replaced by this
        import-only-module revision.";
    }
  }
}

grouping yang-pkg-library-definition {
  description
    "The set of parameters that describe a particular YANG package.";

  uses yang-pkg-identification-leafs;
  uses yang-pkg-common-leafs;

  leaf-list module-set {
    type leafref {
      path "/yanglib:yang-library/yanglib:module-set/yanglib:name";
    }
    description
      "Describes any modules in addition to, and replacing, and
      modules defined in the imported packages.

      If a non import-only module appears in multiple module sets,
      then the module revision and the associated features and
      deviations must be identical.";
  }
}
}
<CODE ENDS>

```

<CODE BEGINS> file "ietf-yang-package2018-11-26.yang"

```

module ietf-yang-package {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-yang-package";
  prefix pkg;

```



```

import ietf-yang-package-types {
    prefix pkg-types;
    reference "RFC XXX: YANG Schema Versioning.";
}

organization
    "IETF NETMOD (Network Modeling) Working Group";

contact
    "WG Web:  <http://tools.ietf.org/wg/netmod/>
    WG List:  <mailto:netmod@ietf.org>

    Author:   Rob Wilton
              <mailto:rwilton@cisco.com>";

description
    "This module provides a definition of a YANG package, which is
    used as the schema for an YANG instance data document specifying
    a YANG package.

    Copyright (c) 2018 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 Simplified BSD License
    set forth in Section 4.c of the IETF Trust's Legal Provisions
    Relating to IETF Documents
    (http://trustee.ietf.org/license-info).

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

// RFC Ed.: update the date below with the date of RFC publication
// and remove this note.
// RFC Ed.: replace XXXX with actual RFC number and remove this
// note.
revision 2018-11-26 {
    description
        "Initial revision";
    reference
        "RFC XXXX: YANG Schema Versioning.";
}

```

```

/*
 * Top-level container
 */

container yang-package {
  config false;
  description
    "Defines a YANG package.

    Intended to be used to specify a YANG package as an instance
    data document.";

  uses pkg-types:yang-pkg-file-definition;
}
}
<CODE ENDS>

```

```

<CODE BEGINS> file "ietf-yang-library-packages@2018-11-26.yang"
module ietf-yang-library-packages {
  yang-version 1.1;
  namespace
    "urn:ietf:params:xml:ns:yang:ietf-yang-library-packages";
  prefix pkg;

  import ietf-yang-package-types {
    prefix pkg-types;
    reference "RFC XXX: YANG Packages.";
  }
  import ietf-yang-library {
    prefix yanglib;
    reference "RFC 7895bis: YANG Library";
  }

  organization
    "IETF NETMOD (Network Modeling) Working Group";

  contact
    "WG Web:  <http://tools.ietf.org/wg/netmod/>
    WG List:  <mailto:netmod@ietf.org>

    Author:   Rob Wilton
              <mailto:rwilton@cisco.com>";

  description
    "This module provides defined augmentations to YANG library to
    allow a server to report YANG package information.

```

Copyright (c) 2018 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 Simplified BSD License set forth in [Section 4.c](#) of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

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

```
// RFC Ed.: update the date below with the date of RFC publication
// and remove this note.
// RFC Ed.: replace XXXX with actual RFC number and remove this
// note.
revision 2018-11-26 {
  description
    "Initial revision";
  reference
    "RFC XXXX: YANG Schema Versioning.";
}

/*
 * Add in the list of packaged into YANG library.
 */
augment "/yanglib:yang-library" {
  description "Add YANG package definitions into YANG library";

  list package {
    config "false";
    key "name version";

    description "Defines the packages available on this server.";

    uses "pkg-types:yang-pkg-library-definition";
  }
}

/*
```

```

    * Allow schema to be related to a YANG package.
    */
    augment "/yanglib:yang-library/yanglib:schema" {
        description
            "Allow datastore schema to be related to a YANG package";
    }

```

```

    container package {
        leaf name {
            type leafref {
                path "/yanglib:yang-library/package/name";
            }
            description
                "The name of the package this schema relates to.";
        }
        leaf version {
            type leafref {
                path '/yanglib:yang-library/'
                    + 'package[name = current()/../name]/version';
            }
            description
                "The version of the package this schema relates to.";
        }
        description
            "Describes which package the schema directly relates to, if
            any.";
    }
}

/*
 * Allow import-only modules to list the versions that they are
 * replacing.
 */

augment
    "/yanglib:yang-library/yanglib:module-set/" +
    "yanglib:import-only-module" {

        description
            "Add replaces-revision to import-only-module definitions";
    }
}

```

```

leaf-list replaces-revision {
  type yanglib:revision-identifier;
  description
    "Gives the revision of an import-only-module defined in an
    imported package that is replaced by this import-only-module
    revision.

    Only used for YANG package definitions";
}
}
}
<CODE ENDS>

```

## [10.](#) Open Questions/Issues

1. Should the YANG library changes be done as an augmentation (as per this draft), or is a new version of YANG library better?
2. Is it OK for the YANG definition used for package instance data vs YANG library to differ? The reason for the difference is to allow "Module Sets" to be reused, potentially minimizing duplicate data in YANG library.
3. Is name sufficient to uniquely identify a package, or should they also define the equivalent to a namespace? The current proposed solution is for the first entry in the location list to define a canonical location
4. Is disabling features using deviations sufficient? There are some cases where this cannot work, e.g. a deviation cannot remove an identity.
5. Should a package (or implementation) be able to remove modules from a package? Current thinking is the answer should be no because it greatly reduces the usefulness of package conformance.
6. Should a package include [RFC 8199](#) related metadata? E.g., does a package contain device or service level YANG models? The current proposal is to gain this flexibility by allowing module tags to be added to package definitions.

7. Considering version selection, should the YANG library package definition have a flag to indicate whether a particular package can be selected by a client? Probably the answer here is to defer this issue to a separate version selection draft that can add a flag via augmentation.
8. TODO - Once [draft-verdt-netmod-yang-semver](#) is published, add appropriate references to this draft for module level semantic versioning.

## 11. Security Considerations

The YANG modules specified in this document defines a schema for data that is accessed by network management protocols such as NETCONF [[RFC6241](#)] or RESTCONF [[RFC8040](#)]. The lowest NETCONF layer is the secure transport layer, and the mandatory-to-implement secure transport is Secure Shell (SSH) [[RFC6242](#)]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [[RFC5246](#)].

The NETCONF access control model [[RFC6536](#)] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

Similarly to YANG library [[I-D.ietf-netconf-rfc7895bis](#)], some of the readable data nodes in these YANG modules may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes.

One additional key different to YANG library, is that the 'ietf-yang-package' YANG module defines a schema to allow YANG packages to be defined in YANG instance data documents, that are outside the security controls of the network management protocols. Hence, it is important to also consider controlling access to these package instance data documents to restrict access to sensitive information.

As per the YANG library security considerations, the module, revision and version information in YANG packages may help an attacker identify the server capabilities and server implementations with

known bugs since the set of YANG modules supported by a server may reveal the kind of device and the manufacturer of the device. Server vulnerabilities may be specific to particular modules, module revisions, module features, or even module deviations. For example, if a particular operation on a particular data node is known to cause a server to crash or significantly degrade device performance, then the packaging information will help an attacker identify server implementations with such a defect, in order to launch a denial-of-service attack on the device.

## [12.](#) IANA Considerations

It is expected that a central registry of standard YANG package definitions is required to support this packaging solution.

It is unclear whether an IANA registry is also required to manage specific package versions. It is highly desirable to have a specific canonical location, under IETF control, where the definitive YANG package versions can be obtained from.

## [13.](#) References

### [13.1.](#) Normative References

Wilton	Expires June 23, 2019	[Page 27]
--------	-----------------------	-----------

---

Internet-Draft	YANG Packages	December 2018
----------------	---------------	---------------

[I-D.ietf-netconf-rfc7895bis]

Bierman, A., Bjorklund, M., Schoenwaelder, J., Watsen, K., and R. Wilton, "YANG Library", [draft-ietf-netconf-rfc7895bis-07](#) (work in progress), October 2018.

[I-D.ietf-netmod-module-tags]

Hopps, C., Berger, L., and D. Bogdanovic, "YANG Module Tags", [draft-ietf-netmod-module-tags-03](#) (work in progress), October 2018.

[I-D.ietf-netmod-yang-instance-file-format]

Lengyel, B. and B. Claise, "YANG Instance Data File Format", [draft-ietf-netmod-yang-instance-file-format-01](#) (work in progress), December 2018.

- [I-D.verdt-netmod-yang-versioning-reqs]  
Clarke, J., "YANG Module Versioning Requirements", [draft-verdt-netmod-yang-versioning-reqs-02](#) (work in progress), November 2018.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", [RFC 5246](#), DOI 10.17487/RFC5246, August 2008, <<https://www.rfc-editor.org/info/rfc5246>>.
- [RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", [RFC 6241](#), DOI 10.17487/RFC6241, June 2011, <<https://www.rfc-editor.org/info/rfc6241>>.
- [RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", [RFC 6242](#), DOI 10.17487/RFC6242, June 2011, <<https://www.rfc-editor.org/info/rfc6242>>.
- [RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", [RFC 6536](#), DOI 10.17487/RFC6536, March 2012, <<https://www.rfc-editor.org/info/rfc6536>>.
- [RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", [RFC 7950](#), DOI 10.17487/RFC7950, August 2016, <<https://www.rfc-editor.org/info/rfc7950>>.

- [RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", [RFC 8040](#), DOI 10.17487/RFC8040, January 2017, <<https://www.rfc-editor.org/info/rfc8040>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.



- [RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", [RFC 8342](#), DOI 10.17487/RFC8342, March 2018, <<https://www.rfc-editor.org/info/rfc8342>>.

### [13.2.](#) Informative References

- [I-D.bierman-netmod-yang-package]  
Bierman, A., "The YANG Package Statement", [draft-bierman-netmod-yang-package-00](#) (work in progress), July 2015.
- [I-D.ietf-netmod-artwork-folding]  
Watsen, K., Wu, Q., Farrel, A., and B. Claise, "Handling Long Lines in Artwork in Internet-Drafts and RFCs", [draft-ietf-netmod-artwork-folding-00](#) (work in progress), November 2018.
- [openconfigsemver]  
"Semantic Versioning for Openconfig Models",  
<<http://www.openconfig.net/docs/semver/>>.
- [RFC8199] Bogdanovic, D., Claise, B., and C. Moberg, "YANG Module Classification", [RFC 8199](#), DOI 10.17487/RFC8199, July 2017, <<https://www.rfc-editor.org/info/rfc8199>>.

### [Appendix A.](#) Tree output for ietf-yang-library with package augmentations

Complete tree output for ietf-yang-library with package augmentations.

```
module: ietf-yang-library
  +--ro yang-library
  |   +--ro module-set* [name]
  |   |   +--ro name string
  |   |   +--ro module* [name]
  |   |   |   +--ro name yang:yang-identifier
  |   |   |   +--ro revision? revision-identifier
  |   |   |   +--ro namespace inet:uri
```

```

| | | +--ro location*      inet:uri
| | | +--ro submodule* [name]
| | | | +--ro name        yang:yang-identifier
| | | | +--ro revision?   revision-identifier
| | | | +--ro location*   inet:uri
| | | +--ro feature*      yang:yang-identifier
| | | +--ro deviation*    -> ../../module/name
| | +--ro import-only-module* [name revision]
| | | +--ro name          yang:yang-identifier
| | | +--ro revision      union
| | | +--ro namespace     inet:uri
| | | +--ro location*     inet:uri
| | | +--ro submodule* [name]
| | | | +--ro name        yang:yang-identifier
| | | | +--ro revision?   revision-identifier
| | | | +--ro location*   inet:uri
| | | +--ro pkg:replaces-revision*
| | | | yanglib:revision-identifier
+--ro schema* [name]
| +--ro name              string
| +--ro module-set*      -> ../../module-set/name
| +--ro pkg:package
| | +--ro pkg:name?
| | | -> /yanglib:yang-library/package/name
| | +--ro pkg:version?   leafref
+--ro datastore* [name]
| +--ro name              ds:datastore-ref
| +--ro schema            -> ../../schema/name
+--ro content-id          string
+--ro pkg:package* [name version]
| +--ro pkg:name          yang:yang-identifier
| +--ro pkg:version        yang-sem-ver
| +--ro pkg:revision-date? yanglib:revision-identifier
| +--ro pkg:location*     inet:uri
| +--ro pkg:description?  string
| +--ro pkg:reference?     string
| +--ro pkg:tag*          tags:tag
| +--ro pkg:mandatory-feature* string
| +--ro pkg:imported-packages* [name version]
| | +--ro pkg:name        yang:yang-identifier
| | +--ro pkg:version      yang-sem-ver
| +--ro pkg:module-set*
| | -> /yanglib:yang-library/module-set/name
x--ro modules-state
| x--ro module-set-id      string
| x--ro module* [name revision]
| | x--ro name              yang:yang-identifier
| | x--ro revision          union

```

```

+--ro schema?                inet:uri
x--ro namespace              inet:uri
x--ro feature*               yang:yang-identifier
x--ro deviation* [name revision]
|   x--ro name                yang:yang-identifier
|   x--ro revision            union
x--ro conformance-type       enumeration
x--ro submodule* [name revision]
    x--ro name                yang:yang-identifier
    x--ro revision            union
+--ro schema?                inet:uri
```

notifications:

```

+---n yang-library-update
|   +--ro content-id -> /yang-library/content-id
x---n yang-library-change
    x--ro module-set-id -> /modules-state/module-set-id
```

## [Appendix B](#). Examples

This section provides various examples of YANG packages, and as such this text is non-normative. The purpose of the examples is to only illustrate the file format of YANG packages, and how package dependencies work. It does not imply that such packages will be defined by IETF, or which modules would be included in those packages even if they were defined.

### [B.1](#). Example IETF Network Device YANG package

This section provides an instance data document example of an IETF Network Device YANG package formatted in JSON.

This example package is intended to represent the standard set of YANG modules, with import dependencies, to implement a basic network device without any dynamic routing or layer 2 services. E.g., it includes functionality such as system information, interface and basic IP configuration.

As for all YANG packages, all import dependencies are fully resolved. Because this example uses YANG modules that have been standardized before YANG semantic versioning, they modules are referenced by revision date rather than version number.

<CODE BEGINS> file "example-ietf-network-device-pkg.json"  
===== NOTE: '\\\'' line wrapping per BCP XX (RFC XXXX) =====

Wilton

Expires June 23, 2019

[Page 31]

Internet-Draft

YANG Packages

December 2018

```
{
  "ietf-yang-instance-data:instance-data-set": {
    "name": "example-ietf-network-device-pkg",
    "target-ptr": "TBD",
    "timestamp": "2018-12-13T17:00:00Z",
    "description": "Example IETF network device YANG package definiti\
\ion",
    "content-data": {
      "ietf-yang-package:yang-package": {
        "name": "example-ietf-network-device",
        "version": "1.1.2",
        "namespace": "urn:ietf:params:xml:ns:yang-pkg:ietf-network-d\
\evice",
        "location": "file://example.org/yang/packages/ietf-network-d\
\evice@v1.1.2.json",
        "description": "This package defines a small sample set of Y\
\ANG modules that could represent the basic set of modules that a st\
\andard network device might be expected to support.",
        "reference": "XXX, draft-rwilton-netmod-yang-packages",
        "revision-date": "2018-11-26",
        "module": [
          {
            "name": "iana-crypt-hash",
            "revision": "2014-08-06",
            "namespace": "urn:ietf:params:xml:ns:yang:iana-crypt-has\
\h",
            "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/iana-crypt-hash%402014-08-06.yang"
          },
          {
            "name": "ietf-system",
            "revision": "2014-08-06",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-system",
            "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-system%402014-08-06.yang"
          },
        ]
      }
    }
  }
}
```

```

    {
      "name": "ietf-interfaces",
      "revision": "2018-02-20",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-interface\
\s",
      "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-interfaces%402018-02-20.yang"
    },
    {
      "name": "ietf-netconf-acm",
      "revision": "2018-02-14",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-netconf-a\

```

```

\cm",
      "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-netconf-acm%402018-02-14.yang"
    },
    {
      "name": "ietf-key-chain",
      "revision": "2017-06-15",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-key-chain\
",
      "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-key-chain@2017-06-15.yang"
    },
    {
      "name": "ietf-ip",
      "revision": "2018-02-22",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-ip",
      "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-ip%402018-02-22.yang"
    }
  ],
  "import-only-module": [
    {
      "name": "ietf-yang-types",
      "revision": "2013-07-15",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-yang-type\
\s",
      "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-yang-types%402013-07-15.yang"
    },

```

```

        {
            "name": "ietf-inet-types",
            "revision": "2013-07-15",
            "namespace": "urn:ietf:params:xml:ns:yang:ietf-inet-type\
\s",
            "location": "https://raw.githubusercontent.com/YangModel\
\s/yang/master/standard/ietf/RFC/ietf-inet-types%402013-07-15.yang"
        }
    ]
}
}
}
}
<CODE ENDS>

```

## [B.2.](#) Example IETF Basic Routing YANG package

This section provides an instance data document example of a basic IETF Routing YANG package formatted in JSON.

This example package is intended to represent the standard set of YANG modules, with import dependencies, that builds upon the example-ietf-network-device YANG package to add support for basic dynamic routing and ACLs.

As for all YANG packages, all import dependencies are fully resolved. Because this example uses YANG modules that have been standardized before YANG semantic versioning, they modules are referenced by revision date rather than version number. Locations have been excluded where they are not currently known, e.g., for YANG modules defined in IETF drafts. In a normal YANG package, locations would be expected to be provided for all YANG modules.

```

<CODE BEGINS> file "example-ietf-routing-pkg.json"
===== NOTE: '\\\ ' line wrapping per BCP XX (RFC XXXX) =====

```

```

{
  "ietf-yang-instance-data:instance-data-set": {
    "name": "example-ietf-routing-pkg",
    "target-ptr": "TBD",
    "timestamp": "2018-12-13T17:00:00Z",
    "description": "Example IETF routing YANG package definition",
    "content-data": {
      "ietf-yang-package:yang-package": {
        "name": "example-ietf-routing",
        "version": "1.3.1",
        "namespace": "urn:ietf:params:xml:ns:yang-pkg:ietf-routing",
        "location": "file://example.org/yang/packages/ietf-routing@v\
1.3.1.json",
        "description": "This package defines a small sample set of I\
ETF routing YANG modules that could represent the set of IETF routi\
ng functionality that a basic IP network device might be expected t\
o support.",
        "reference": "XXX, draft-rwilton-netmod-yang-packages",
        "revision-date": "2018-11-26",
        "imported-packages": [
          {
            "name": "ietf-network-device",
            "version": "1.1.2",
            "location": [
              "http://example.org/yang/packages/ietf-network-device@\
v1.1.2.json"

```

```

    ]
  }
],
"module": [
  {
    "name": "ietf-routing",
    "revision": "2018-03-13",
    "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing",
    "location": [
      "https://raw.githubusercontent.com/YangModels/yang/master/standard/ietf/RFC/ietf-routing@2018-03-13.yang"
    ]
  },
  {
    "name": "ietf-ipv4-unicast-routing",

```

```

        "revision": "2018-03-13",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ipv4-unca\
\st-routing",
        "location": [
            "https://raw.githubusercontent.com/YangModels/yang/master/standard/ietf/RFC/ietf-ipv4-unicast-routing@2018-03-13.yang"
        ]
    },
    {
        "name": "ietf-ipv6-unicast-routing",
        "revision": "2018-03-13",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ipv6-unca\
\st-routing",
        "location": [
            "https://raw.githubusercontent.com/YangModels/yang/master/standard/ietf/RFC/ietf-ipv6-unicast-routing@2018-03-13.yang"
        ]
    },
    {
        "name": "ietf-isis",
        "revision": "2018-12-11",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-isis"
    },
    {
        "name": "ietf-interfaces-common",
        "revision": "2018-07-02",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-interface\
\s-common"
    },
    {
        "name": "ietf-if-l3-vlan",
        "revision": "2017-10-30",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-if-l3-vla\
"
    },
    {
        "name": "ietf-routing-policy",
        "revision": "2018-10-19",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing-p\
\olicy"
    },

```



```

    {
      "name": "ietf-bgp",
      "revision": "2018-05-09",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-bgp"
    },
    {
      "name": "ietf-access-control-list",
      "revision": "2018-11-06",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-access-co\
\ntrl-list"
    }
  ],
  "import-only-module": [
    {
      "name": "ietf-routing-types",
      "revision": "2017-12-04",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-routing-t\
\nypes",
      "location": [
        "https://raw.githubusercontent.com/YangModels/yang/master/standard/ietf/RFC/ietf-routing-types@2017-12-04.yang"
      ]
    },
    {
      "name": "iana-routing-types",
      "revision": "2017-12-04",
      "namespace": "urn:ietf:params:xml:ns:yang:iana-routing-t\
\nypes",
      "location": [
        "https://raw.githubusercontent.com/YangModels/yang/master/standard/ietf/RFC/iana-routing-types@2017-12-04.yang"
      ]
    },
    {
      "name": "ietf-bgp-types",
      "revision": "2018-05-09",
      "namespace": "urn:ietf:params:xml:ns:yang:ietf-bgp-types"
    },
    {
      "name": "ietf-packet-fields",

```

```

        "namespace": "urn:ietf:params:xml:ns:yang:ietf-packet-fi\
\elds"
    },
    {
        "name": "ietf-ethertypes",
        "revision": "2018-11-06",
        "namespace": "urn:ietf:params:xml:ns:yang:ietf-ethertype\
\s"
    }
]
}
}
}
}
<CODE ENDS>

```

### [B.3.](#) Package import conflict resolution example

This section provides an example of how a package can resolve conflicting module versions from imported packages.

In this example, YANG package 'example-3-pkg' imports both 'example-import-1' and 'example-import-2' packages. However, the two imported packages implement different versions of 'example-module-A' so the 'example-3-pkg' package selects version '1.2.3' to resolve the conflict. Similarly, for import-only modules, the 'example-3-pkg' package does not require both versions of example-types-module-C to be imported, so it indicates that it only imports revision '2018-11-26' and not '2018-01-01'.

```

{
  "ietf-yang-instance-data:instance-data-set": {
    "name": "example-import-1-pkg",
    "description": "First imported example package",
    "content-data": {
      "ietf-yang-package:yang-package": {
        "name": "example-import-1",
        "version": "1.0.0",
        "reference": "XXX, draft-rwilton-netmod-yang-packages",
        "revision-date": "2018-01-01",
        "module": [
          {
            "name": "example-module-A",
            "version": "1.0.0"
          },
        ],
      },
    },
  },
}

```

```
{
  {
    "name": "example-module-B",
    "version": "1.0.0"
  }
],
"import-only-module": [
  {
    "name": "example-types-module-C",
    "revision": "2018-01-01"
  },
  {
    "name": "example-types-module-D",
    "revision": "2018-01-01"
  }
]
}
}
}
}

{
  "ietf-yang-instance-data:instance-data-set": {
    "name": "example-import-2-pkg",
    "description": "Second imported example package",
    "content-data": {
      "ietf-yang-package:yang-package": {
        "name": "example-import-2",
        "version": "2.0.0",
        "reference": "XXX, draft-rwilton-netmod-yang-packages",
        "revision-date": "2018-11-26",
        "module": [
          {
            "name": "example-module-A",
            "version": "1.2.3"
          },
          {
            "name": "example-module-E",
            "version": "1.1.0"
          }
        ],
        "import-only-module": [
          {
            "name": "example-types-module-C",
            "revision": "2018-11-26"
          },
          {
```

```
"name": "example-types-module-D",
"revision": "2018-11-26"
```

```
    }
  ]
}
}
}
{
  "ietf-yang-instance-data:instance-data-set": {
    "name": "example-3-pkg",
    "description": "Importing example package",
    "content-data": {
      "ietf-yang-package:yang-package": {
        "name": "example-3",
        "version": "1.0.0",
        "reference": "XXX, draft-rwilton-netmod-yang-packages",
        "revision-date": "2018-11-26",
        "imported-packages": [
          {
            "name": "example-import-1",
            "version": "1.0.0"
          },
          {
            "name": "example-import-2",
            "version": "2.0.0"
          }
        ],
        "module": [
          {
            "name": "example-module-A",
            "version": "1.2.3"
          }
        ],
        "import-only-module": [
          {
            "name": "example-types-module-C",
            "revision": "2018-11-26",
            "replaces-revision": [ "2018-01-01 " ]
          }
        ]
      }
    }
  }
}
```

```
}  
}  
}  
}  
}
```

Author's Address

Wilton	Expires June 23, 2019	[Page 39]
--------	-----------------------	-----------

---

Internet-Draft	YANG Packages	December 2018
----------------	---------------	---------------

Robert Wilton  
Cisco Systems, Inc.

Wilton

Expires June 23, 2019

[Page 40]