Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 3 September 2025

R. Wills
Cisco Systems

2 March 2025

Yang Templates
draft-wills-netmod-yang-templates-00

Abstract

Yang Templates is a way to more easily manage a network device's configuration when parts of that configuration are repetitive. This document provides a high-level description of Yang Templates, and is expected to evolve into a full solution in later versions.

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 3 September 2025.

Table of Contents

1. Introduction

This document describes a mechanism whereby nodes of configuration data can be placed into templates, and templates can be applied to subtrees in a configuration datastore. When a template is applied to a subtree, the configuration in the template takes effect for that subtree (unless other configuration takes precedence, as described later in this document).

The contents of this document are based on discussions in two Interim Meetings of the Netmod working group. It was written _after_ those meetings and is intended to summarise the outcomes as a set of high-level requirements. It is not an exhaustive set of minutes.

It's expected that later versions of this document will contain a fuller description of Yang Templates.

1.1. Requirements Language

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.

2. Overview

The configuration of network devices is modeled using the Yang [RFC7950] language, and the configuration of a specific device is determined by the contents of one or more configuration datastores on the device.

A configuration datastore often contains multiple subtrees with similar or identical nodes within them. For example, a device may be configured to have multiple BGP neighbors. Each neighbor could have an identical set of properties (achieved using identical configuration on each neighbor), and neighbor-specific properties such as the address (achieved using configuration that varies between neighbors).

In this situation, a client must ensure that each set of identical nodes exists in all the relevant subtrees of the datastore and that each occurrence of a node has the same value in order to achieve the correct intended configuration. This creates problems in keeping occurrences of these nodes in sync and understanding the contents of a large configuration datastore.

Yang Templates defines fragments of configuration called "templates". When an implementation supports Yang Templates, the client is able to use existing network management protocols such as NETCONF [RFC6241] to:

* Manage the configuration inside each template.

* Apply templates to subtrees of the datastore.

The configuration from templates and the configuration not in templates is combined together to produce an intended datastore that describes the configuration of the device.

3. Requirements

This section describes the requirements that the Yang Templates solution must satisfy. These requirements were all discussed in the Interim Meetings, and a rough consensus was reached on each of them by the participants in the meetings.

A general theme of the Yang Templates work is to come up with a "Minimal Viable Product" that is useful but not over-complicated. More advanced features could be considered as extensions in later drafts.

3.1. Defining and managing templates

Templates can be used with any Yang module. They contain nodes of configuration data, and are stored persistently in the running datastore of the device.

A client can view and manipulate a template, including the configuration inside it, by manipulating it in the <running> datastore. In this sense, a template and its contents behaves like any other subtree of configuration.

3.2. Applying templates

A template can be applied to zero or more nodes in the <running> datastore. Each node can have zero or more templates applied to it, and the order they are applied is specified by the client. The order is important when determining the final intended configuration -- see the next section.

Templates can be applied at multiple points in the hierachy. The next section states the requirements when a node applies a template and it has an ancestor that also applies a template.

When viewing the <running> datastore, there is a mechanism to see which templates have been applied to each node, and in which order.

3.3. Producing the <intended> datastore

The device’s <intended> datastore is the result of combining all the applications of templates together with non-template config. This is called "expanding out" the templates.

The intended configuration inside a subtree is the result of taking the relevant contents of every template applied to the subtree’s root node and its ancestors, and combining it with the (non-template) data nodes inside the subtree.

A node inside a subtree may be present in multiple templates that have been applied, and/or it may be present as non-template config inside the subtree. The requirements for combining the templates and the non-template config together are as follows:

* The value of a node in the <intended> configuration is determined by using precedence to decide where to take the value from.

* Non-template config always has the highest precedence.

* When templates are applied to multiple ancestors, the innermost ancestor takes precedence.

* When multiple templates are applied to a particular node, the order of application (as indicated by the client when applying the templates) determines the precedence within that node.

Whenever the contents of a template is updated in <running>, the result of expanding out the template appears in <intended> and takes effect on the device.

3.4. Pattern matching in templates

The configuration inside a template definition can contain values for list keys that are simple regular expressions, using a limited subset of regular expression syntax. This controls which list entries that subtree of the template takes effect for when it is applied.

An example of this would be to have a template that is applied to a top-level <interfaces> container, but the template only takes effect for certain interface names that match the regular expression.

3.5. Off-box template expansion

If the client knows the contents of the <running> datastore (non-template config, template definitions and template applications), it must be possible for the client to calculate the result of template expansion.

In other words, the outcome of template expansion depends solely on the <running> datastore and not the state of the device.

4. IANA Considerations

This memo includes no request to IANA.

5. Security Considerations

This document should not affect the security of the Internet.

6. References

6.1. Normative References

[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>.

[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>.

[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>.

[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>.

6.2. Informative References

Acknowledgements

The author would like to thank Shiya Ashraf, Lou Berger, Quifang Ma, Robert Peschi, Deepak Rajaram, Jason Sterne, Kent Watsen and Robert Wilton for comments and contributions made during interim meetings.

The author would like to acknowledge the following drafts and presenters for kick-starting discussions on Yang Templates:

* draft-ma-netmod-yang-config-template-00

* draft-rajaram-netmod-yang-cfg-template-framework-00

* Jan Lindblad

Author’s Address

Robert Wills
Cisco Systems
Email: rowills@cisco.com