Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 21, 2016

R. Shakir
Jive Communications

October 19, 2015

# A Data Model for Network Instances in Service Provider Networks
## draft-openconfig-rtgwg-network-instance-00

Abstract

This document defines a YANG data model for configuring and managing forwarding instances on a network device - focused primarily on deployments in service provider networks. The model is vendor-neutral and based on operational requirements expressed by operators. A 'network instance' is defined to be a discrete forwarding table on a network element, which contains Layer 3 and/or Layer 2 forwarding entries. Each network instance may instantiate its own sets of protocols which control installation of forwarding entries into one or more tables (which may be Layer 2 FIBs, or Layer 3 RIBs).

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 http://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 April 21, 2016.

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## 1. Introduction

Within service provider networks, it is typical for a network element to maintain multiple forwarding tables, allowing the network to support multiple topologies. For instance, a service provider may offer Layer 2 or Layer 3 VPN services to its customers - which requires a forwarding table per customer to be maintained on a network element. In other cases, multiple L3 services may be deployed to allow services such as Internet connectivity and private topologies to be maintained on a device.

This YANG [RFC6020] data model defines a construct which can be used to configure and retrieve operational state related to such instances.

### 1.1. Goals and Approach

The concept of maintaining multiple Layer 2 virtual switch instances (VSIs) and/or multiple Layer 3 virtual routing and forwarding (VRF) tables is common on network elements that are deployed in service provider networks. However, within such networks there are requirements for a network instance to run both Layer 2 and Layer 3 routing protocols, or contain routed interfaces (e.g., a VPLS service where there is a Layer 3 interface within it, for subscriber termination or to act as a default gateway). For this reason, the intention of the model presented here is to define a generic construct which allows isolation of a forwarding table which may contain Layer 3 RIB entries, Layer 2 FIB entries, or a combination both. This differs to the approach taken in other models, which tend to focus on solely L2 VSIs or L3 VRFs. An instance within the model is referred to as network instance - and may be configured to support L3 RIB entries only (i.e., be functionally equivalent to a VRF), L2 FIB entries only (i.e., act as a VSI) or support a combination of both.

The model presented in this document is explicitly biased towards deployments on service provider network equipment, as discussed between members of the OpenConfig project.

## 2. Model overview

```
module: openconfig-network-instance
   +--rw network-instances
      +--rw network-instance* [name]
         +--rw name                        -> ../config/name
         +--rw config
            ...
         +--ro state
            ...
         +--rw inter-instance-policies
         |  +--rw apply-policy
            ...
         +--rw table-connections
         |  +--rw table-connection* [src-table dst-table]
         |     +--rw src-table       -> ../config/src-table
         |     +--rw dst-table       -> ../config/dst-table
                ...
         |     +--rw apply-policy
                ...
         +--rw tables
         |  +--rw table* [table-name]
                ...
         +--rw interfaces
                ....
         +--rw connection-points
         |  +--rw connection-point* [connection-point-id]
              ...
         +--rw protocols
            +--rw protocol* [identifier name]
                ...
               +--rw static
                ...
               +--rw aggregate
                ...
```

The key elements of the network instance model shown in the above outline tree view are:

- Type of network instance and key configuration parameters that relate to it (e.g., configuration of a route distinguisher for L3 routing instance).

- Policies which define how the instance exchanges routes with other instances - for example, allowing forwarding entries to be leaked between a global and private network instance.

- The individual tables that are maintained by a network instance. This caters for the approach whereby an implementation maintains per-protocol tables - and it is possible to examine these tables separately. The choice to implement this table definition is an open issue - as discussed in Section 3.1. Along with this table definition, a set of policies determining how entries are leaked between the tables is defined.

- Connections between the different tables that are maintained by the network instance. Where protocols populate individual tables, such connections facilitate leaking of routes between individual protocols (by means of populating entries from one into the table of the other).

- The interfaces associated with a certain network instance - the current implementation assumption (as demonstrated in the openconfig-interfaces models) is that configuration parameters that relate directly to the interface (e.g., IP addressing) are configured directly under the interface construct. This requires that a network instance must have a means to be able to associate itself with an interface.

- A logical grouping of local or remote interfaces into 'connection points' utilised by a number of Layer 2 forwarding approaches (e.g., redundancy for attachment points for PWE or L2VPN services). This allows interfaces which are associated with the network instance to be associated with one another, and hence referenced as a group. This construct can be used to represent redundant interfaces for a P2P L2VPN - and is extensible to support Ethernet segments for other L2VPN types.

- A list of the protocols that are instantiated under the network instance. The protocols that are defined in [I-D.openconfig-rtgwg-local-routing] are included directly such that forwarding entries generated as static or aggregate routes can be matched in a routing policy. Other protocols, such as BGP (for which the data model is defined in [I-D.ietf-idr-bgp-model] would augment this list to include their own configuration options). It should be noted that the list of protocols is not keyed directly on the protocol name, but rather also includes an identifier for that instance. This supports cases where a single network instance may run multiple instances of a protocol.

It is expected that all devices maintain at least one network instance, which represents the default (or global) forwarding table of the device. Additional instances are intended to support virtual instances, such as VSIs and VRFs.

Where base configuration is required for such instances to be utilised (e.g., an route distinguisher value is required) then this is configured globally for each network instance. It expected that additional parameters, such as the encapsulation utilised for the instance will be added in future revisions of the model.

Policies for leaking of routes between instances and tables leverage [I-D.ietf-rtgwg-policy-model] as a policy framework. This allows definition of 'import' policies (allowing a pull model) or export policies (allowing a push model) to be utilised for installation of entries into a local or remote network instance. The same framework is utilised to leak prefixes between multiple tables that are instantiated under an individual network instance.

### 2.1. Supporting Layer 2 Forwarding Constructs

Whilst the initial focus of parameters in the model is to allow the instantiation of attributes related to Layer 3 forwarding constructs - it is expected that other configuration parameters, such as the definition of 'cross-connections' between local and/or remote interfaces or connection points can be utilised to represent forwarding constructs within the network instance.

The addition of this to the model is still a work-in-progress. The inclusion of the connection point configuration and state parameters is intended to demonstrate how L2 and L3 constructs can co-exist with one another within the model.

## 3. Open Issues

### 3.1. Representation of per-protocol tables vs. a single table for all protocols

Some implementations maintain a single table into which all protocols install entries. Each protocol is then configured to explicitly import entries from that table into their local 'export' table. Others maintain multiple RIBs and require explicit connections to be made between these tables such that entries from one protocol are made available to another.

There are essentially two alternatives for how this is modelled in the network-instance model:

1. Allow each protocol to specify a 'target-table' that indicates the table that its entries are to be installed into. In the case that the same target table is specified for multiple protocols, implicit import/export configuration between tables could be created on implementations that implement multiple RIBs.

Explicit configuration can be utilised to explicitly allow an operator to leak between tables in the case that multiple target tables are specified. It should be noted that in fact, protocols that support multiple address families need to allow specification of the target table on a per-address-family basis.

2. Assume that all protocols implement their own target tables. In this case, explicit configuration is utilised to leak between them. Implementations that do not support such per-protocol tables could implicitly generate the configuration that results in forwarding entries being leaked between the tables such that the appearance of a single target-table is maintained.

As per the discussion in Section 2 currently, this model chooses the former implementation approach.

## 4. Security Considerations

Routing configuration has a significant impact on network operations, and as such any related model carries potential security risks.

YANG data models are generally designed to be used with the NETCONF protocol over an SSH transport. This provides an authenticated and secure channel over which to transfer configuration and operational data. Note that use of alternate transport or data encoding (e.g., JSON over HTTPS) would require similar mechanisms for authenticating and securing access to configuration data.

Most of the data elements in the network intsance model could be considered sensitive from a security standpoint. Unauthorized access or invalid data could cause major disruption.

## 5. IANA Considerations

This YANG data model and the component modules currently use a temporary ad-hoc namespace. If and when it is placed on redirected for the standards track, an appropriate namespace URI will be registered in the IETF XML Registry" [RFC3688]. The routing policy YANG modules will be registered in the "YANG Module Names" registry [RFC6020].

## 6. YANG module

The network instance model is described by the YANG module below.

<CODE BEGINS> file openconfig-network-instance.yang
<?yfile include="../../yang/network-instance/openconfig-network-instance.yang"?
>
<CODE ENDS>

<CODE BEGINS> file openconfig-network-instance-types.yang
<?yfile include="../../yang/network-instance/openconfig-network-instance-types.yang"?>
<CODE ENDS>

## 7. References

### 7.1. Normative references

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[I-D.openconfig-rtgwg-local-routing] Shaikh, A., George, J., Shakir, R., and F. Chen, "A Data Model for Locally Generated Routes in Service Provider Networks", draft-openconfig-rtgwg-local-routing-00 (work in progress), July 2015.

[I-D.ietf-rtgwg-policy-model] Shaikh, A., rjs@rob.sh, r., D'Souza, K., and C. Chase, "Routing Policy Configuration Model for Service Provider Networks", draft-ietf-rtgwg-policy-model-00 (work in progress), September 2015.

[I-D.ietf-idr-bgp-model] Shaikh, A., Shakir, R., Patel, K., Hares, S., D'Souza, K., Bansal, D., Clemm, A., Alex, A., Jethanandani, M., and X. Liu, "BGP Model for Service Provider Networks", draft-ietf-idr-bgp-model-00 (work in progress), July 2015.

### 7.2. Informative references

[YANG-REPO] "OpenConfig YANG Data Model Repository - Network Instance", October 2015, <https://github.com/openconfig/public/releases/network-

instance>.

## Appendix A. Acknowledgements

TODO

Author's Address

Rob Shakir
Jive Communications, Inc.
1275 West 1600 North, Suite 100
Orem, UT 84057

Email: rjs@rob.sh