Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 23, 2017

X. Liu
Jabil
Y. Qu
Futurewei Technologies, Inc.
A. Lindem
Cisco Systems
C. Hopps
Deutsche Telekom
L. Berger
LabN Consulting, L.L.C.

February 19, 2017

Routing Area Common YANG Data Types
draft-ietf-rtgwg-routing-types-01

Abstract

This document defines a collection of common data types using the YANG data modeling language. These derived common types are designed to be imported by other modules defined in the routing area.

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 August 23, 2017.

Copyright Notice

Table of Contents

1. Introduction

The YANG [RFC6020] [RFC7950] is a data modeling language used to model configuration data, state data, Remote Procedure Calls, and notifications for network management protocols. The YANG language supports a small set of built-in data types and provides mechanisms to derive other types from the built-in types.

This document introduces a collection of common data types derived from the built-in YANG data types. The derived types are designed to be the common types applicable for modeling in the routing area.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in BCP 14, RFC 2119 [RFC2119].

1.2. Terminology

The terminology for describing YANG data models is found in [RFC7950].

2. Overview

This document defines the following data types:

router-id
: Router Identifiers are commonly used to identify a nodes in routing and other control plane protocols. An example usage of router-id can be found in [I-D.ietf-ospf-yang].

address-family
: This type defines values for use in address family identifiers. The values are based on the IANA Address Family Numbers Registry [1]. An example usage can be found in [I-D.ietf-idr-bgp-model].

route-target
: Route Targets (RTs) are commonly used to control the distribution of virtual routing and forwarding (VRF) information, see [RFC4364], in support of virtual private networks (VPNs). An example usage can be found in [I-D.ietf-bess-l2vpn-yang].

route-target-type
: This type defines the import and export rules of Route Targets, as descibed in Section 4.3.1 of [RFC4364]. An example usage can be found in [I-D.ietf-idr-bgp-model].

route-distinguisher
: Route Distinguishers (RDs) are commonly used to identify separate routes in support of virtual private networks (VPNs). For example, in [RFC4364], RDs are commonly used to identify independent VPNs and VRFs, and more generally, to identify multiple routes to the same prefix. An example usage can be found in [I-D.ietf-idr-bgp-model].

ipv4-multicast-group-address
: This type defines the representation of an IPv4 multicast group address, which is in the range from 224.0.0.0 to 239.255.255.255. An example usage can be found in [I-D.ietf-pim-yang].

ipv6-multicast-group-address
: This type defines the representation of an IPv6 multicast group address, which is in the range of FF00::/8. An example usage can be found in [I-D.ietf-pim-yang].

ip-multicast-group-address
: This type represents an IP multicast group address and is IP version neutral. The format of the textual representation implies the IP version. An example usage can be found in [I-D.ietf-pim-yang].

ipv4-multicast-source-address
IPv4 source address type for use in multicast control protocols. This type also allows the indication of wildcard sources, i.e., "*". An example of where this type may/will be used is [I-D.ietf-pim-yang].

ipv6-multicast-source-address
IPv6 source address type for use in multicast control protocols. This type also allows the indication of wildcard sources, i.e., "*". An example of where this type may/will be used is [I-D.ietf-pim-yang].

ieee-bandwidth
Bandwidth in IEEE 754 floating point 32-bit binary format [IEEE754]. Commonly used in Traffic Engineering control plane protocols. An example of where this type may/will be used is [I-D.ietf-ospf-yang].

link-access-type
This type identifies the IGP link type. An example of where this type may/will be used is [I-D.ietf-ospf-yang].

timer-multiplier
This type is used in conjunction with a timer-value type. It is generally used to indicate define the number of timer-value intervals that may expire before a specific event must occur. Examples of this include the arrival of any BFD packets, see [RFC5880] Section 6.8.4, or hello_interval in [RFC3209]. Example of where this type may/will be used is [I-D.ietf-idr-bgp-model] and [I-D.ietf-teas-yang-rsvp].

timer-value-seconds16
This type covers timers which can be set in seconds, not set, or set to infinity. This type supports a range of values that can be represented in a uint16 (2 octets). An example of where this type may/will be used is [I-D.ietf-ospf-yang].

timer-value-seconds32
This type covers timers which can be set in seconds, not set, or set to infinity. This type supports a range of values that can be represented in a uint32 (4 octets). An example of where this type may/will be used is [I-D.ietf-teas-yang-rsvp].

timer-value-milliseconds
This type covers timers which can be set in milliseconds, not set, or set to infinity. This type supports a range of values that can be represented in a uint32 (4 octets). Examples of where this

type may/will be used include [I-D.ietf-teas-yang-rsvp] and [I-D.ietf-bfd-yang].

generalized-label
This type represents a generalized label for Generalized Multi-Protocol Label Switching (GMPLS) [RFC3471]. The Generalized Label does not identify its type, which is known from the context. An example usage can be found in [I-D.ietf-teas-yang-te].

mpls-label-special-purpose
This type represents the special-purpose Multiprotocol Label Switching (MPLS) label values [RFC7274]. An example usage can be found in [I-D.ietf-mpls-base-yang].

mpls-label-general-use
The 20 bits label values in an MPLS label stack entry, specified in [RFC3032]. This label value does not include the encodings of Traffic Class and TTL (time to live). The label range specified by this type is for general use, with special-purpose MPLS label values excluded. An example usage can be found in [I-D.ietf-mpls-base-yang].

mpls-label
The 20 bits label values in an MPLS label stack entry, specified in [RFC3032]. This label value does not include the encodings of Traffic Class and TTL (time to live). The label range specified by this type covers the general use values and the special-purpose label values. An example usage can be found in [I-D.ietf-mpls-base-yang].

This document defines the following YANG groupings:

mpls-label-stack
This grouping defines a reusable collection of schema nodes representing an MPLS label stack [RFC3032]. An example usage can be found in [I-D.ietf-mpls-base-yang].

vpn-route-target-rules
This grouping defines a reusable collection of schema nodes representing Route Target import-export rules used in the BGP enabled Virtual Private Networks (VPNs). [RFC4364][RFC4664]. An example usage can be found in [I-D.ietf-bess-l2vpn-yang].

3. YANG Module

```
<CODE BEGINS> file "ietf-routing-types@2017-02-19.yang"
module ietf-routing-types {
```

```
namespace "urn:ietf:params:xml:ns:yang:ietf-routing-types";
prefix "rt-types";

import ietf-yang-types {
  prefix "yang";
}

import ietf-inet-types {
  prefix "inet";
}

organization "IETF Routing Area Working Group (rtgwg)";

contact
    "Routing Area Working Group - <rtgwg@ietf.org>";

description
  "This module contains a collection of YANG data types
   considered generally useful for routing protocols.";

revision 2017-02-19 {
  description
    "Initial revision.";
  reference
    "RFC TBD: Routing YANG Data Types";
}

/*** collection of types related to routing ***/
typedef router-id {
  type yang:dotted-quad;
  description
    "A 32-bit number in the dotted quad format assigned to each
     router. This number uniquely identifies the router within an
     Autonomous System.";
}

// address-family
identity address-family {
  description
    "Base identity from which identities describing address
     families are derived.";
}

identity ipv4 {
  base address-family;
  description
    "This identity represents IPv4 address family.";
}
```

```
     identity ipv6 {
       base address-family;
       description
         "This identity represents IPv6 address family.";
     }

     //The rest of the values deinfed in the IANA registry

     identity nsap {
       base address-family;
       description
         "Address family from IANA registry.";
     }
     identity hdlc {
       base address-family;
       description
         "(8-bit multidrop)
           Address family from IANA registry.";
     }
     identity bbn1822 {
       base address-family;
       description
         "AHIP (BBN report #1822)
          Address family from IANA registry.";
     }
     identity ieee802 {
       base address-family;
       description
         "(includes all 802 media plus Ethernet canonical format)
          Address family from IANA registry.";
     }
     identity e163 {
       base address-family;
       description
         "Address family from IANA registry.";
     }
     identity e164 {
       base address-family;
       description
         "SMDS, Frame Relay, ATM
          Address family from IANA registry.";
     }
     identity f69 {
       base address-family;
       description
         "(Telex)
          Address family from IANA registry.";
     }
```

```
    identity x121 {
      base address-family;
      description
        "(X.25, Frame Relay)
         Address family from IANA registry.";
    }
    identity ipx {
      base address-family;
      description
        "Address family from IANA registry.";
    }
    identity appletalk {
      base address-family;
      description
        "Address family from IANA registry.";
    }
    identity decnet-iv {
      base address-family;
      description
        "Decnet IV
         Address family from IANA registry.";
    }
    identity vines {
      base address-family;
      description
        "Banyan Vines
         Address family from IANA registry.";
    }
    identity e164-nsap {
      base address-family;
      description
        "E.164 with NSAP format subaddress
         Address family from IANA registry.";
    }
    identity dns {
      base address-family;
      description
        "Domain Name System
         Address family from IANA registry.";
    }
    identity dn {
      base address-family;
      description
        "Distinguished Name
         Address family from IANA registry.";
    }
    identity as-num {
      base address-family;
```

```
      description
        "AS Number
         Address family from IANA registry.";
    }
    identity xtp-v4 {
      base address-family;
      description
        "XTP over IPv4
         Address family from IANA registry.";
    }
    identity xtp-v6 {
      base address-family;
      description
        "XTP over IPv6
         Address family from IANA registry.";
    }
    identity xtp {
      base address-family;
      description
        "XTP native mode XTP
         Address family from IANA registry.";
    }
    identity fc-port {
      base address-family;
      description
        "Fibre Channel World-Wide Port Name
         Address family from IANA registry.";
    }
    identity fc-node {
      base address-family;
      description
        "Fibre Channel World-Wide Node Name
         Address family from IANA registry.";
    }
    identity gwid {
      base address-family;
      description
        "Address family from IANA registry.";
    }
    identity l2vpn {
      base address-family;
      description
        "Address family from IANA registry.";
    }
    identity mpls-tp-section-eid {
      base address-family;
      description
        "MPLS-TP Section Endpoint Identifier
```

```
          Address family from IANA registry.";
      }
      identity mpls-tp-lsp-eid {
        base address-family;
        description
          "MPLS-TP LSP Endpoint Identifier
           Address family from IANA registry.";
      }
      identity mpls-tp-pwe-eid {
        base address-family;
        description
          "MPLS-TP Pseudowire Endpoint Identifier
           Address family from IANA registry.";
      }
      identity mt-v4 {
        base address-family;
        description
          "Multi-Topology IPv4.
           Address family from IANA registry.";
      }
      identity mt-v6 {
        base address-family;
        description
          "Multi-Topology IPv6.
           Address family from IANA registry.";
      }

      /*** collection of types related to VPN ***/
      typedef route-target {
        type string {
          pattern
            '(0:(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|'
          + '[0-5]?\d{0,3}\d):(429496729[0-5]|42949672[0-8]\d|'
          + '4294967[01]\d{2}|429496[0-6]\d{3}|42949[0-5]\d{4}|'
          + '4294[0-8]\d{5}|429[0-3]\d{6}|42[0-8]\d{7}|4[01]\d{8}|'
          + '[0-3]?\d{0,8}\d))|'
          + '(1:(((\d|[1-9]\d|1\d{2}|2[0-4]\d|25[0-5])\.){3}(\d|[1-9]\d|'
          + '1\d{2}|2[0-4]\d|25[0-5])):(6553[0-5]|655[0-2]\d|'
          + '65[0-4]\d{2}|6[0-4]\d{3}|[0-5]?\d{0,3}\d))|'
          + '(2:(429496729[0-5]|42949672[0-8]\d|4294967[01]\d{2}|'
          + '429496[0-6]\d{3}|42949[0-5]\d{4}|4294[0-8]\d{5}|'
          + '429[0-3]\d{6}|42[0-8]\d{7}|4[01]\d{8}|[0-3]?\d{0,8}\d):'
          + '(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|'
          + '[0-5]?\d{0,3}\d))';
        }
        description
          "A route target is an 8-octet BGP extended community
           initially identifying a set of sites in a BGP
```

```
          VPN (RFC 4364). However, it has since taken on a more
          general role in BGP route filtering.
          A route target consists of three fields:
          a 2-octet type field, an administrator field,
          and an assigned number field.
          According to the data formats for type 0, 1, and 2 defined in
          RFC4360 and RFC5668, the encoding pattern is defined as:

          0:2-octet-asn:4-octet-number
          1:4-octet-ipv4addr:2-octet-number
          2:4-octet-asn:2-octet-number.

          Some valid examples are: 0:100:100, 1:1.1.1.1:100, and
          2:1234567890:203.";
        reference
          "RFC4360: BGP Extended Communities Attribute.
           RFC5668: 4-Octet AS Specific BGP Extended Community.";
      }

      typedef route-target-type {
        type enumeration {
          enum "import" {
            value "0";
            description
              "The route target applies to route import.";
           }
          enum "export" {
            value "1";
            description
              "The route target applies to route export.";
           }
          enum "both" {
            value "2";
            description
              "The route target applies to both route import and
               route export.";
           }
        }
        description
          "Indicates the role a route target takes
           in route filtering.";
        reference
          "RFC4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
      }

      typedef route-distinguisher {
        type string {
          pattern
```

```
          '(0:(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|'
        + '[0-5]?\d{0,3}\d):(429496729[0-5]|42949672[0-8]\d|'
        + '4294967[01]\d{2}|429496[0-6]\d{3}|42949[0-5]\d{4}|'
        + '4294[0-8]\d{5}|429[0-3]\d{6}|42[0-8]\d{7}|4[01]\d{8}|'
        + '[0-3]?\d{0,8}\d))|'
        + '(1:(((\d|[1-9]\d|1\d{2}|2[0-4]\d|25[0-5])\.){3}(\d|[1-9]\d|'
        + '1\d{2}|2[0-4]\d|25[0-5])):(6553[0-5]|655[0-2]\d|'
        + '65[0-4]\d{2}|6[0-4]\d{3}|[0-5]?\d{0,3}\d))|'
        + '(2:(429496729[0-5]|42949672[0-8]\d|4294967[01]\d{2}|'
        + '429496[0-6]\d{3}|42949[0-5]\d{4}|4294[0-8]\d{5}|'
        + '429[0-3]\d{6}|42[0-8]\d{7}|4[01]\d{8}|[0-3]?\d{0,8}\d):'
        + '(6553[0-5]|655[0-2]\d|65[0-4]\d{2}|6[0-4]\d{3}|'
        + '[0-5]?\d{0,3}\d))|'
        + '(([3-9a-fA-F]|[1-9a-fA-F][\da-fA-F]{1,3}):'
        + '[\da-fA-F]{1,12})';
      }
      description
        "A route distinguisher is an 8-octet value used to distinguish
         routes from different BGP VPNs (RFC 4364). A route
         distinguisher consists of three fields: A 2-octet type field,
         an administrator field, and an assigned number field.
         According to the data formats for type 0, 1, and 2 defined in
         RFC4364, the encoding pattern is defined as:

         0:2-octet-asn:4-octet-number
         1:4-octet-ipv4addr:2-octet-number
         2:4-octet-asn:2-octet-number.
         2-octet-other-hex-number:6-octet-hex-number

         Some valid examples are: 0:100:100, 1:1.1.1.1:100, and
         2:1234567890:203.";
      reference
        "RFC4364: BGP/MPLS IP Virtual Private Networks (VPNs).";
    }

    /*** collection of types common to multicast ***/
    typedef ipv4-multicast-group-address {
      type inet:ipv4-address {
        pattern '(2((2[4-9])|(3[0-9]))\.).*';
      }
      description
        "This type represents an IPv4 multicast group address,
         which is in the range from 224.0.0.0 to 239.255.255.255.";
      reference
        "RFC1112: Host Extensions for IP Multicasting.";
    }

    typedef ipv6-multicast-group-address {
```

```
      type inet:ipv6-address {
        pattern
          '(([fF]{2}[0-9a-fA-F]{2}):).*';
      }
      description
        "This type represents an IPv6 multicast group address,
         which is in the range of FF00::/8.";
      reference
        "RFC4291: IP Version 6 Addressing Architecture. Sec 2.7.
         RFC7346: IPv6 Multicast Address Scopes.";
    }

    typedef ip-multicast-group-address {
      type union {
        type ipv4-multicast-group-address;
        type ipv6-multicast-group-address;
      }
      description
       "This type represents an IP multicast group address and is IP
        version neutral. The format of the textual representation
        implies the IP version.";
    }

    typedef ipv4-multicast-source-address {
      type union {
        type enumeration {
          enum '*' {
            description
            "Any source address.";
          }
        }
        type inet:ipv4-address;
      }
      description
        "Multicast source IPv4 address type.";
    }

    typedef ipv6-multicast-source-address {
      type union {
        type enumeration {
          enum '*' {
            description
            "Any source address.";
          }
        }
        type inet:ipv6-address;
      }
      description
```

```
        "Multicast source IPv6 address type.";
      }

      /*** collection of types common to protocols ***/
      typedef ieee-bandwidth {
        type string {
          pattern
            '0[xX](0((\.0?)?[pP](\+)?0?|(\.0?))|'
          + '1(\.([\da-fA-F]{0,5}[02468aAcCeE]?)?)?[pP](\+)?(12[0-7]|'
          + '1[01]\d|0?\d?\d)?)|0[xX][\da-fA-F]{1,8}';
        }
        description
          "Bandwidth in IEEE 754 floating point 32-bit binary format:
           (-1)**(S) * 2**(Exponent-127) * (1 + Fraction),
           where Exponent uses 8 bits, and Fraction uses 23 bits.
           The units are octets per second.
           The encoding format is the external hexadecimal-significand
           character sequences specified in IEEE 754 and C99,
           restricted to be normalized, non-negative, and non-fraction:
           0x1.hhhhhhp{+}d or 0X1.HHHHHHP{+}D
           where 'h' and 'H' are hexadecimal digits, 'd' and 'D' are
           integers in the range of [0..127].
           When six hexadecimal digits are used for 'hhhhhh' or 'HHHHHH',
           the least significant digit must be an even number.
           'x' and 'X' indicate hexadecimal; 'p' and 'P' indicate power
           of two.
           Some examples are: 0x0p0, 0x1p10, and 0x1.abcde2p+20";
        reference
          "IEEE Std 754-2008: IEEE Standard for Floating-Point
           Arithmetic.";
      }

      typedef link-access-type {
        type enumeration {
          enum "broadcast" {
            description
            "Specify broadcast multi-access network.";
          }
          enum "non-broadcast-multiaccess" {
            description
            "Specify Non-Broadcast Multi-Access (NBMA) network.";
          }
          enum "point-to-multipoint" {
            description
            "Specify point-to-multipoint network.";
          }
          enum "point-to-point" {
            description
```

```
          "Specify point-to-point network.";
        }
      }
      description
        "Link access type.";
    }

    typedef timer-multiplier {
      type uint8;
      description
        "The number of timer value intervals that should be
         interpreted as a failure.";
    }

    typedef timer-value-seconds16 {
      type union {
        type uint16 {
          range "1..65535";
          }
        type enumeration {
          enum "infinity" {
            description "The timer is set to infinity.";
          }
          enum "not-set" {
            description "The timer is not set.";
          }
        }
      }
      units seconds;
      description "Timer value type, in seconds (16 bit range).";
    }

    typedef timer-value-seconds32 {
      type union {
        type uint32 {
          range "1..4294967295";
          }
        type enumeration {
          enum "infinity" {
            description "The timer is set to infinity.";
          }
          enum "not-set" {
            description "The timer is not set.";
          }
        }
      }
      units seconds;
      description "Timer value type, in seconds (32 bit range).";
```

```
    }

    typedef timer-value-milliseconds {
      type union {
        type uint32{
          range "1..4294967295";
          }
        type enumeration {
          enum "infinity" {
            description "The timer is set to infinity.";
          }
          enum "not-set" {
            description "The timer is not set.";
          }
        }
      }
      units milliseconds;
      description "Timer value type, in milliseconds.";
    }

    /*** collection of types related to MPLS/GMPLS ***/
    typedef generalized-label {
      type binary;
      description
        "Generalized label. Nodes sending and receiving the
         Generalized Label know the kinds of link they are
         using. Hence, the Generalized Label does not identify
         its type.  Instead, nodes are expected to know from
         the context and type of label to expect.";
      reference "RFC3471: Section 3.2";
    }

    identity mpls-label-special-purpose-value {
      description
        "Base identity for deriving identities describing
         special-purpose Multiprotocol Label Switching (MPLS) label
         values.";
      reference
        "RFC7274: Allocating and Retiring Special-Purpose MPLS
         Labels.";
    }

    identity ipv4-explicit-null-label {
      base mpls-label-special-purpose-value;
      description
        "This identity represents the IPv4 Explicit NULL Label.";
      reference
        "RFC3032: MPLS Label Stack Encoding. Section 2.1.";
```

```
  }

  identity router-alert-label {
    base mpls-label-special-purpose-value;
    description
      "This identity represents the Router Alert Label.";
    reference
      "RFC3032: MPLS Label Stack Encoding. Section 2.1.";
  }

  identity ipv6-explicit-null-label {
    base mpls-label-special-purpose-value;
    description
      "This identity represents the IPv6 Explicit NULL Label.";
    reference
      "RFC3032: MPLS Label Stack Encoding. Section 2.1.";
  }

  identity implicit-null-label {
    base mpls-label-special-purpose-value;
    description
      "This identity represents the Implicit NULL Label.";
    reference
      "RFC3032: MPLS Label Stack Encoding. Section 2.1.";
  }

  identity entropy-label-indicator {
    base mpls-label-special-purpose-value;
    description
      "This identity represents the Entropy Label Indicator.";
    reference
      "RFC6790: The Use of Entropy Labels in MPLS Forwarding.
       Sections 3 and 10.1.";
  }

  identity gal-label {
    base mpls-label-special-purpose-value;
    description
      "This identity represents the Generic Associated Channel Label
       (GAL).";
    reference
      "RFC5586: MPLS Generic Associated Channel.
       Sections 4 and 10.";
  }

  identity oam-alert-label {
    base mpls-label-special-purpose-value;
    description
```

```
        "This identity represents the OAM Alert Label.";
      reference
        "RFC3429: Assignment of the 'OAM Alert Label' for Multiprotocol
         Label Switching Architecture (MPLS) Operation and Maintenance
         (OAM) Functions.
         Sections 3 and 6.";
    }

    identity extension-label {
      base mpls-label-special-purpose-value;
      description
        "This identity represents the Extension Label.";
      reference
        "RFC7274: Allocating and Retiring Special-Purpose MPLS Labels.
         Sections 3.1 and 5.";
    }

    typedef mpls-label-special-purpose {
      type identityref {
        base mpls-label-special-purpose-value;
      }
      description
        "This type represents the special-purpose Multiprotocol Label
         Switching (MPLS) label values.";
      reference
        "RFC3032: MPLS Label Stack Encoding.
         RFC7274: Allocating and Retiring Special-Purpose MPLS
         Labels.";
    }

    typedef mpls-label-general-use {
      type uint32 {
        range "16..1048575";
      }
      description
        "The 20 bits label values in an MPLS label stack entry,
         specified in RFC3032. This label value does not include
         the encodings of Traffic Class and TTL (time to live).
         The label range specified by this type is for general use,
         with special-purpose MPLS label values excluded.";
      reference
        "RFC3032: MPLS Label Stack Encoding.";
    }

    typedef mpls-label {
      type union {
        type mpls-label-special-purpose;
        type mpls-label-general-use;
```

```
      }
      description
        "The 20 bits label values in an MPLS label stack entry,
         specified in RFC3032. This label value does not include
         the encodings of Traffic Class and TTL (time to live).";
      reference
        "RFC3032: MPLS Label Stack Encoding.";
    }

    /*
     * Groupings
     */
    grouping mpls-label-stack {
      description
        "A grouping that specifies an MPLS label stack.";
      container mpls-label-stack {
        description
          "Container for a list of MPLS label stack entries.";
        list entry {
          key "id";
          description
            "List of MPLS label stack entries.";
          leaf id {
            type uint8;
            description
              "Identifies the sequence of an MPLS label stack entries.
               An entry with smaller ID value is precedes an entry in
               the label stack with a smaller ID.";
          }
          leaf label {
            type rt-types:mpls-label;
            description
              "Label value.";
          }
          leaf ttl {
            type uint8;
            description
              "Time to Live (TTL).";
            reference
              "RFC3032: MPLS Label Stack Encoding.";
          }
          leaf traffic-class {
            type uint8 {
              range "0..7";
            }
            description
              "Traffic Class (TC).";
            reference
```

```
              "RFC5462: Multiprotocol Label Switching (MPLS) Label
               Stack Entry: 'EXP' Field Renamed to 'Traffic Class'
               Field.";
          }
        }
      }
    } // mpls-label-stack

    grouping vpn-route-targets {
      description
        "A grouping that specifies Route Target import-export rules
         used in the BGP enabled Virtual Private Networks (VPNs).";
      reference
        "RFC4364: BGP/MPLS IP Virtual Private Networks (VPNs).
         RFC4664: Framework for Layer 2 Virtual Private Networks
         (L2VPNs)";
      list vpn-target {
        key route-target;
        description
          "List of Route Targets.";
        leaf route-target {
          type rt-types:route-target;
          description
            "Route Target value";
        }
        leaf route-target-type {
          type rt-types:route-target-type;
          mandatory true;
          description
            "Import/export type of the Route Target.";
        }
      }
    } // vpn-route-targets
  }
  <CODE ENDS>
```

4. IANA Considerations

RFC Ed.: In this section, replace all occurrences of 'XXXX' with the actual RFC number (and remove this note).

This document registers the following namespace URIs in the IETF XML registry [RFC3688]:

```
---------------------------------------------------------------------
URI: urn:ietf:params:xml:ns:yang:ietf-routing-types
Registrant Contact: The IESG.
XML: N/A, the requested URI is an XML namespace.
---------------------------------------------------------------------
```

This document registers the following YANG modules in the YANG Module Names registry [RFC6020]:

```
---------------------------------------------------------------------
name:         ietf-routing-types
namespace:    urn:ietf:params:xml:ns:yang:ietf-routing-types
prefix:       rt-types
reference:    RFC XXXX
---------------------------------------------------------------------
```

5. Security Considerations

This document defines common data types using the YANG data modeling language. The definitions themselves have no security impact on the Internet, but the usage of these definitions in concrete YANG modules might have. The security considerations spelled out in the YANG specification [RFC7950] apply for this document as well.

6. Acknowledgements

The Routing Area Yang Architecture design team members included Acee Lindem, Anees Shaikh, Christian Hopps, Dean Bogdanovic, Ebben Aries, Lou Berger, Qin Wu, Rob Shakir, Xufeng Liu, and Yingzhen Qu.

7. References

7.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[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>.

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

7.2. Informative References

[IEEE754] IEEE, "IEEE Standard for Floating-Point Arithmetic", IEEE Std 754-2008, August 2008.

[I-D.ietf-bfd-yang]
Zheng, L., Rahman, R., Networks, J., Jethanandani, M., and G. Mirsky, "Yang Data Model for Bidirectional Forwarding Detection (BFD)", draft-ietf-bfd-yang-04 (work in progress), January 2017.

[I-D.ietf-idr-bgp-model]
Shaikh, A., Shakir, R., Patel, K., Hares, S., D'Souza, K., Bansal, D., Clemm, A., Zhdankin, A., Jethanandani, M., and X. Liu, "BGP Model for Service Provider Networks", draft-ietf-idr-bgp-model-02 (work in progress), July 2016.

[I-D.ietf-ospf-yang]
Yeung, D., Qu, Y., Zhang, Z., Chen, I., and A. Lindem, "Yang Data Model for OSPF Protocol", draft-ietf-ospf-yang-06 (work in progress), October 2016.

[I-D.ietf-pim-yang]
Liu, X., McAllister, P., Peter, A., Sivakumar, M., Liu, Y., and f. hu, "A YANG data model for Protocol-Independent Multicast (PIM)", draft-ietf-pim-yang-03 (work in progress), October 2016.

[I-D.ietf-teas-yang-rsvp]
Beeram, V., Saad, T., Gandhi, R., Liu, X., Shah, H., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for Resource Reservation Protocol (RSVP)", draft-ietf-teas-yang-rsvp-06 (work in progress), October 2016.

[I-D.ietf-teas-yang-te]
Saad, T., Gandhi, R., Liu, X., Beeram, V., Shah, H., Bryskin, I., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for Traffic Engineering Tunnels and Interfaces", draft-ietf-teas-yang-te-05 (work in progress), October 2016.

[I-D.ietf-bess-l2vpn-yang]
Shah, H., Brissette, P., Chen, I., Hussain, I., and B. Wen, "YANG Data Model for MPLS-based L2VPN", draft-ietf-bess-l2vpn-yang-02 (work in progress), October 2016.

[I-D.ietf-mpls-base-yang]
Raza, K., Gandhi, R., Liu, X., Beeram, V., Saad, T., Bryskin, I., Chen, X., Jones, R., and B. Wen, "A YANG Data Model for MPLS Base", draft-ietf-mpls-base-yang-01 (work in progress), July 2016.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, DOI 10.17487/RFC3032, January 2001, <http://www.rfc-editor.org/info/rfc3032>.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, DOI 10.17487/RFC3209, December 2001, <http://www.rfc-editor.org/info/rfc3209>.

[RFC3471] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", RFC 3471, DOI 10.17487/RFC3471, January 2003, <http://www.rfc-editor.org/info/rfc3471>.

[RFC4364] Rosen, E. and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, DOI 10.17487/RFC4364, February 2006, <http://www.rfc-editor.org/info/rfc4364>.

[RFC4664] Andersson, L., Ed. and E. Rosen, Ed., "Framework for Layer 2 Virtual Private Networks (L2VPNs)", RFC 4664, DOI 10.17487/RFC4664, September 2006, <http://www.rfc-editor.org/info/rfc4664>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <http://www.rfc-editor.org/info/rfc5880>.

[RFC7274] Kompella, K., Andersson, L., and A. Farrel, "Allocating and Retiring Special-Purpose MPLS Labels", RFC 7274, DOI 10.17487/RFC7274, June 2014, <http://www.rfc-editor.org/info/rfc7274>.

7.3. URIs

[1] http://www.iana.org/assignments/address-family-numbers/address-family-numbers.xhtml

Authors' Addresses

Xufeng Liu
Jabil
8281 Greensboro Drive, Suite 200
McLean VA 22102
USA

EMail: Xufeng_Liu@jabil.com

Yingzhen Qu
Futurewei Technologies, Inc.
2330 Central Expressway
Santa Clara CA 95050
USA

EMail: yingzhen.qu@huawei.com

Acee Lindem
Cisco Systems
301 Midenhall Way
Cary, NC 27513
USA

EMail: acee@cisco.com

Christian Hopps
Deutsche Telekom

EMail: chopps@chopps.org

Lou Berger
LabN Consulting, L.L.C.

EMail: lberger@labn.net