Babel Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 24, 2019

M. Jethanandani
VMware
B. Stark
AT&T

October 21, 2018

# YANG Data Model for Babel
# draft-mahesh-babel-yang-model-00

## Abstract

This document defines a data model for the Babel routing protocol. The data model is defined using the YANG data modeling language.

## 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 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here..

## 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 April 24, 2019.

## Copyright Notice

Table of Contents



## 1. Introduction

This document defines a data model for the Babel routing protocol [I-D.ietf-babel-rfc6126bis]. The data model is defined using the YANG [RFC7950] data modeling language. It is based on the Babel Information Model [I-D.ietf-babel-information-model].

Artwork in this document contains shorthand references to drafts in progress. Please apply the following replacements

- "XXXX" --> the assigned RFC value for this draft both in this draft and in the YANG models under the revision statement.

- Revision date in model, in the format 2018-04-27 needs to get updated with the date the draft gets approved. The date also needs to get reflected on the line with <CODE BEGINS>.

## 1.1. Definitions and Acronyms

- o





## 1.2. Tree Diagram

For a reference to the annotations used in tree diagrams included in this draft, please see YANG Tree Diagrams [RFC8340].

# 2. Babel Module

This document defines a YANG 1.1 [RFC7950] data model for the configuration and management of Babel. The YANG module is based on the Babel Information Model [I-D.ietf-babel-information-model].

## 2.1. Information Model

## 2.2. YANG Module

This module imports definitions from Common YANG Data Types [RFC6991].

```
module: ietf-babel
  +--rw babel!
     +--rw version?                  string
     +--rw enable?                   boolean
     +--rw router-id                 binary
     +--rw link-type*                identityref
     +--ro sequence-number?          yang:counter32
     +--rw cost-compute-algorithm*   identityref
     +--rw security-supported*       identityref
     +--rw transport
     |  +--rw udp-port?      inet:port-number
     |  +--rw mcast-group?   inet:ip-address
     +--rw interfaces* [reference]
     |  +--rw reference                if:interface-ref
     |  +--rw enable?                  boolean
     |  +--rw link-type?               identityref
     |  +--ro mcast-hello-seqno?       int16
     |  +--ro ucast-hello-seqno?       int16
```

```
|  +--ro mcast-hello-interval?   int16
|  +--ro ucast-hello-interval?   int16
|  +--rw update-interval?        uint32
|  +--rw external-cost?          uint32
|  +--rw message-log-enable?     boolean
|  +--rw message-log* [log-time]
|  |  +--rw log-time     yang:timestamp
|  |  +--rw log-entry?   string
|  +--rw neighbor-objects* [neighbor-address]
|  |  +--rw neighbor-address         inet:ip-address
|  |  +--rw hello-mcast-history?     string
|  |  +--rw hello-ucast-history?     string
```





```
|  |  +--rw txcost?                  int32
|  |  +--rw exp-mcast-hello-seqno?   int32
|  |  +--rw exp-ucast-hello-seqno?   int32
|  |  +--rw neighbor-ihu-interval?   int32
|  |  +--rw rxcost?                  int32
|  |  +--rw cost?                    int32
|  +--rw security* [mechanism]
|     +--rw mechanism               string
|     +--rw enable?                 boolean
|     +--rw self-cred* [id]
|     |  +--rw id      string
|     |  +--rw cred?   binary
|     +--rw trust* [id]
|     |  +--rw id      string
|     |  +--rw cred?   binary
|     +--rw credvalid-log-enable?   boolean
|     +--rw credvalid-log* [log-time]
|        +--rw log-time     yang:timestamp
|        +--rw log-entry?   string
+--rw routes* [prefix]
|  +--rw prefix                   inet:ip-address
|  +--rw prefix-length?           inet:ip-prefix
|  +--rw router-id?               binary
|  +--rw neighbor?
|  |       -> ../../interfaces/neighbor-objects/neighbor-address
|  +--rw (metric)
|  |  +--:(received-metric)
|  |  |  +--rw received-metric?    int32
|  |  +--:(calcuated-metric)
```

```
      |  |     +--rw calcuated-metric?   int32
      |  +--rw seqno?                    int32
      |  +--rw next-hop?                 inet:ip-address
      |  +--rw feasible?                 boolean
      |  +--rw selected?                 boolean
      +--rw security* [mechanism]
         +--rw mechanism               string
         +--rw enable?                 boolean
         +--rw self-cred* [id]
         |  +--rw id      string
         |  +--rw cred?   binary
         +--rw trust* [id]
         |  +--rw id      string
         |  +--rw cred?   binary
         +--rw credvalid-log-enable?   boolean
         +--rw credvalid-log* [log-time]
            +--rw log-time     yang:timestamp
            +--rw log-entry?   string
```





```
  <CODE BEGINS> file "ietf-babel@2018-10-21.yang"

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

    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-interfaces {
      prefix if;
      reference
        "RFC 8343 - A YANG Data Model for Interface Management";
    }
```

```
    organization
      "IETF Babel routing protocl Working Group";

    contact
      "WG Web: http://tools.ietf.org/wg/babel/
       WG List: babel@ietf.org

       Editor: Mahesh Jethanandani
               mjethanandani@gmail.com
       Editor: Barbara Stark
               bs7652@att.com";

    description
      "This YANG module defines a model for the Babel routing
       protocol.

       Copyright (c) 2018 IETF Trust and the persons identified as
       the document authors.  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.";

    revision 2018-10-21 {
      description
        "Initial version.";
      reference
        "RFC XXX: Babel YANG Data Model.";
    }

    /*
     * Identities
     */
    identity babel-link-type {
      description
```

```
        "Base identity from which all Babel Link Types are derived.";
    }

    identity ethernet {
      base "babel-link-type";
      description
        "Ethernet link type for Babel Routing Protocol.";
    }
    identity other {
      base "babel-link-type";
      description
        "Other link type for Babel Routing Protocol.";
    }
    identity tunnel {
      base "babel-link-type";
      description
        "Tunnel link type for Babel Routing Protocol.";
    }
    identity wireless {
      base "babel-link-type";
      description
        "Wireless link type for Babel Routing Protocol.";
    }
    identity moca {
      base "babel-link-type";
      description
        "Multimedia over Coax Alliance.";
    }
    identity g-hn-over-coax {
      base "babel-link-type";
      description
        "G.hn over coax.";
      reference
```





```
        "G.9960: Unified high-speed wireline-base home networking
         transceivers.";
    }
    identity g-hn-over-powerline {
      base "babel-link-type";
      description
        "G.hn over powerline.";
      reference
```

```
        "G.9960: Unified high-speed wireline-base home networking
         transceivers.";
    }
    identity home-plug {
      base "babel-link-type";
      description
        "HomePlug Power Alliance.";
      reference
        "IEEE 1901: HD-PC";
    }
    identity ieee-802-15 {
      base "babel-link-type";
      description
        "Wireless Personal Area Networks (WPAN).";
      reference
        "IEEE 802.15: Wireless Personal Area Networks (WPAN).";
    }

    identity babel-cost-compute-algorithm {
      description
        "Base identity from which all Babel cost compute algorithms
         are derived.";
    }
    identity k-out-of-j {
      base "babel-cost-compute-algorithm";
      description
        "k-out-of-j algorithm.";
    }
    identity etx {
      base "babel-cost-compute-algorithm";
      description
        "Expected Transmission Count.";
    }

    /*
     * Babel type identities
     */
    identity babel-security-supported {
      description
        "Base identity from which all Babel security types are
```





```
         derived.";
```

```
}

/*
 * Features
 */

/*
 * Features supported
 */

/*
 * Typedefs
 */
typedef base64 {
  type string {
    pattern '(([A-Za-z0-9+/]{4})*([A-Za-z0-9+/]{3}=|'
          + '[A-Za-z0-9+/]{2}==)?){1}';
  }
  description
    "A binary-to-text encoding scheme to represent binary data in
     an ASCII string format.";
  reference
    "RFC 4648, The Base16, Base32, and Base64 Data Encodings";
}

/*
 * Groupings
 */
grouping log {
  leaf log-time {
    type yang:timestamp;
    description
      "The date and time (according to the device internal
       clock setting, which may be a time relative to boot
       time, acquired from NTP, configured by the user, etc.)
       when this log entry was created.";
    reference
      "RFC YYYY, Babel Information Model, Section 4.2.";
  }

  leaf log-entry {
    type string;
    description
      "The logged message, as a string of utf-8 encoded hex
       characters.";
    reference
      "RFC YYYY, Babel Information Model, Section 4.2.";
```

```
      }
      description
        "A babel-log-obj list.";
      reference
        "RFC YYYY, Babel Information Model, Section 4.2.";
    }

    grouping credential {
      leaf id {
        type string;
        description
          "An identifier that identifies this credential uniquely.";
      }

      leaf cred {
        type binary;
        description
          "A credential, such as an X.509 certificate, a public key,
           etc. used for signing and/or encrypting babel messages.";
        reference
          "RFC YYYY, Babel Information Model, Section 4.1.";
      }
      description
        "A babel-credential-obj list.";
      reference
        "RFC YYYY, Babel Information Model, Section 4.1.";
    }

    grouping security {
      leaf mechanism {
        type string;
        description
          "The name of the security mechanism this object instance
           is about. The value MUST be the same as one of the
           identities listed as the babel-security-supported
           parameter.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.5.";
      }

      leaf enable {
        type boolean;
        description
          "If true, the security mechanism is running. If false,
           the security mechanism is not currently running.";
        reference
```

```
        "RFC YYYY, Babel Information Model, Section 3.5.";
    }
```





```
    list self-cred {
      key "id";

      uses credential;
      description
        "Credentials this router presents to participate in the
         enabled security mechanism. Any private key component of
         a credential MUST NOT be readable. Adding and deleting
         credentials MAY be allowed.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.5.";
    }

    list trust {
      key "id";

      uses credential;
      description
        "A list of credential-obj objects that identify the
         credentials of routers whose babel messages may be
         trusted or of a certificate authority (CA) whose signing
         of a router's credentials implies the router credentials
         can be trusted, in the context of this security
         mechanism. How a security mechanism interacts with this
         list is determined by the mechanism. A security algorithm
         may do additional validation of credentials, such as
         checking validity dates or revocation lists, so presence
         in this list may not be sufficient to determine trust.
         Adding and deleting credentials MAY be allowed.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.5.";
    }

    leaf credvalid-log-enable {
      type boolean;
      description
        "If true, logging of messages that include credentials
         used for authentication is enabled. If false, these
         messages are not logged.";
```

```
        reference
          "RFC YYYY, Babel Information Model, Section 3.5.";
      }

      list credvalid-log {
        key "log-time";

        uses log;
        description
```





```
          "Log entries that have the timestamp a message containing
           credentials used for peer authentication (e.g., DTLS
           Server Hello) was received on a Babel port, and the
           entire received message (including Ethernet frame and IP
           headers, if possible); an implementation must restrict
           the size of this log, but how and what size is
           implementation-specific.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.5.";
      }
      description
        "A babel-security-obj list.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.5.";
    }

    /*
     * Data model
     */
    container babel {
      presence "A Babel container.";
      description
        "This is a top level container for the Babel routing protocol.";

      leaf version {
        type string;
        description
          "This is the version of the babel protocol implemented.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.1.";
      }
```

```
      leaf enable {
        type boolean;
        default false;
        description
          "When written, it configures whether the protocol should be
           enabled. A read from the <running> or <intended> datastore
           therefore indicates the configured adminstrative value of
           whether the protocol is enabled or not.

           A read from the <operational> datastore indicates whether
           the protocol is actually running or not, i.e. it indicates
           the operational state of the protocol.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.1.";
      }
```





```
      leaf router-id {
        type binary;
        mandatory "true";
        description
          "Every Babel speaker is assigned a router-id, which is an
           arbitrary string of 8 octets that is assumed to be unique
           across the routing domain";
        reference
          "RFC YYYY, Babel Information Model, Section 3.1,
           rfc6126bis, The Babel Routing Protocol. Section 3.";
      }

      leaf-list link-type {
        type identityref {
          base "babel-link-type";
        }
        description
          "Link types supported by this implementation of Babel.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.1.";
      }

      leaf sequence-number {
        type yang:counter32;
        config false;
```

```
      description
        "Sequence number included in route updates for routes
         originated by this node.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.1.";
    }

    leaf-list cost-compute-algorithm {
      type identityref {
        base "babel-cost-compute-algorithm";
      }
      description
        "List of cost compute algorithms supported by this
         implementation of Babel.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.1.";
    }

    leaf-list security-supported {
      type identityref {
        base "babel-security-supported";
      }
      description
```





```
        "Babel security mechanism used by this implementation or
         per interface.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.1.";
    }

    container transport {
      leaf udp-port {
        type inet:port-number;
        default "6696";
        description
          "UDP port for sending and receiving Babel messages. The
           default port is 6696.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.2.";
      }

      leaf mcast-group {
```

```
        type inet:ip-address;
        default "ff02:0:0:0:0:0:1:6";
        description
          "Multicast group for sending and receiving multicast
           announcements on IPv6.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.2.";
      }
      description
        "Babel Transport object.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.1.";
    }
    list interfaces {
      key "reference";

      leaf reference {
        type if:interface-ref;
        description
          "Reference to an interface object as defined by the data
           model (e.g., YANG, BBF TR-181); data model is assumed to
           allow for referencing of interface objects which may be at
           any layer (physical, Ethernet MAC, IP, tunneled IP, etc.).
           Referencing syntax will be specific to the data model. If
           there is no set of interface objects available, this should
           be a string that indicates the interface name used by the
           underlying operating system.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }
```





```
      leaf enable {
        type boolean;
        default "true";
        description
          "If true, babel sends and receives messages on this
           interface. If false, babel messages received on this
           interface are ignored and none are sent.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }
      leaf link-type {
```

```
        type identityref {
          base babel-link-type;
        }
        description
          "Indicates the type of link. Set of values of supported
           link types where the following enumeration values MUST
           be supported when applicable: 'ethernet', 'wireless',
           'tunnel', and 'other'. Additional values MAY be
           supported.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }
      leaf mcast-hello-seqno {
        type int16;
        config false;
        description
          "The current sequence number in use for multicast hellos
           sent on this interface.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }
      leaf ucast-hello-seqno {
        type int16;
        config false;
        description
          "The current sequence number in use for unicast hellos
           sent on this interface.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }
      leaf mcast-hello-interval {
        type int16;
        config false;
        description
          "The current multicast hello interval in use for hellos
           sent on this interface.";
        reference
```





```
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }
      leaf ucast-hello-interval {
        type int16;
```

```
      config false;
      description
        "The current unicast hello interval in use for hellos sent
         on this interface.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.3.";
    }
    leaf update-interval {
      type uint32;
      description
        "The current update interval in use for this interface.";
      reference
    "RFC YYYY, Babel Information Model, Section 3.3.";
    }
    leaf external-cost {
      type uint32;
      description
        "External input to cost of link of this interface. If
         supported, this is a value that is added to the metrics
         of routes learned over this interface. How an
         implementation uses the value is up to the implementation,
         which means the use may not be consistent across
         implementations.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.3.";
    }
    leaf message-log-enable {
      type boolean;
      description
        "If true, logging of babel messages received on this
         interface is enabled; if false, babel messages are not
         logged.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.3.";
    }

    list message-log {
      key "log-time";

      uses log;
      description
        "Log entries that have timestamp of a received Babel
         message and the entire received Babel message (including
         Ethernet frame and IP headers, if possible). An
```

```
           implementation must restrict the size of this log, but how
           and what size is implementation specific.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.3.";
      }

      list neighbor-objects {
        key "neighbor-address";

        leaf neighbor-address {
          type inet:ip-address;
          description
            "IPv4 or v6 address the neighbor sends messages from.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
        }

        leaf hello-mcast-history {
          type string;
          description
            "The multicast Hello history of whether or not the
             multicast Hello messages prior to babel-exp-mcast-
             hello-seqno were received, with a '1' for the most
             recent Hello placed in the most significant bit and
             prior Hellos shifted right (with '0' bits placed
             between prior Hellos and most recent Hello for any
             not-received Hellos); represented as a string using
             utf-8 encoded hex digits where a '1' bit = Hello
             received and a '0' bit = Hello not received.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
        }

        leaf hello-ucast-history {
          type string;
          description
            "The unicast Hello history of whether or not the
             unicast Hello messages prior to babel-exp-ucast-
             hello-seqno were received, with a '1' for the most
             recent Hello placed in the most significant bit and
             prior Hellos shifted right (with '0' bits placed
             between prior Hellos and most recent Hello for any
             not-received Hellos); represented as a string using
             utf-8 encoded hex digits where a '1' bit = Hello
             received and a '0' bit = Hello not received.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
```

```
        }
```





```
        leaf txcost {
          type int32;
          description
            "Transmission cost value from the last IHU packet
             received from this neighbor, or maximum value
             (infinity) to indicates the IHU hold timer for this
             neighbor has expired description.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
        }

        leaf exp-mcast-hello-seqno {
          type int32;
          description
            "Expected multicast Hello sequence number of next Hello
             to be received from this neighbor; if multicast Hello
             messages are not expected, or processing of multicast
             messages is not enabled, this MUST be 0.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
        }

        leaf exp-ucast-hello-seqno {
          type int32;
          description
            "Expected unicast Hello sequence number of next Hello to
             be received from this neighbor; if unicast Hello
             messages are not expected, or processing of unicast
             messages is not enabled, this MUST be 0.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
        }

        leaf neighbor-ihu-interval {
          type int32;
          description
            "Current IHU interval for this neighbor.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.4.";
        }
```

```
          leaf rxcost {
            type int32;
            description
              "Reception cost calculated for this neighbor. This value
               is usually derived from the Hello history, which may be
               combined with other data, such as statistics maintained
               by the link layer. The rxcost is sent to a neighbour in
```





```
               each IHU.";
            reference
              "RFC YYYY, Babel Information Model, Section 3.4.";
          }

          leaf cost {
            type int32;
            description
              "Link cost is computed from the values maintained in
               the neighbour table. The statistics kept in the neighbour
               table about the reception of Hellos, and the txcost
               computed from received IHU packets.";
            reference
              "RFC YYYY, Babel Information Model, Section 3.4.";
          }
          description
            "A set of Babel Neighbor Object.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.3.";
        }

        list security {
          key "mechanism";

          uses security;
          description
            "A security-obj object that applies to this interface. If
             implemented, this allows security to be enabled only on
             specific interfaces or allows different security mechanisms
             to be enabled on different interfaces.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.3.";
        }
```

```
      description
        "A set of Babel Interface objects.";
      reference
        "RFC YYYY, Babel Information Model, Section 3.1.";
    }

    list routes {
      key "prefix";

      leaf prefix {
        type inet:ip-address;
        description
          "Prefix (expressed in IP address format) for which this
           route is advertised.";
        reference
```





```
          "RFC YYYY, Babel Information Model, Section 3.6.";
      }

      leaf prefix-length {
        type inet:ip-prefix;
        description
          "Length of the prefix for which this route is advertised.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.6.";
      }

      leaf router-id {
        type binary;
        description
          "router-id of the source router for which this route is
           advertised.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.6.";
      }

      leaf neighbor {
        type leafref {
          path "../../interfaces/neighbor-objects/neighbor-address";
        }
        description
          "Reference to the babel-neighbors entry for the neighbor
```

```
           that advertised this route.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.6.";
      }

      choice metric {
        mandatory "true";
        leaf received-metric {
          type int32;
          description
            "The metric with which this route was advertised by the
             neighbor, or maximum value (infinity) to indicate a the
             route was recently retracted and is temporarily
             unreachable. this metric will be 0 (zero) if the route
             was not received from a neighbor but was generated
             through other means. Either babel-route-calculated-metric
             or babel-route-received-metric MUST be provided.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.6,
             draft-ietf-babel-rfc6126bis, The Babel Routing Protocol,
             Section 3.5.5.";
        }
```





```
        leaf calcuated-metric {
          type int32;
          description
            "A calculated metric for this route. How the metric is
             calculated is implementation-specific. Maximum value
             (infinity) indicates the route was recently retracted
             and is temporarily unreachable. Either
             babel-route-calculated-metric or
             babel-route-received-metric MUST be provided.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.6,
             draft-ietf-babel-rfc6126bis, The Babel Routing Protocol,
             Section 3.5.5.";
        }
        description
          "Either babel-route-calculated-metric or
           babel-route-received-metric MUST be provided.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.6,
```

```
            draft-ietf-babel-rfc6126bis, The Babel Routing Protocol,
            Section 3.5.5.";
        }

        leaf seqno {
          type int32;
          description
            "The sequence number with which this route was advertised.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.6.";
        }

        leaf next-hop {
          type inet:ip-address;
          description
            "The next-hop address of this route. This will be empty if
             this route has no next-hop address.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.6.";
        }

        leaf feasible {
          type boolean;
          description
            "A boolean flag indicating whether this route is feasible.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.6,
             draft-ietf-babel-rfc6126bis, The Babel Routing Protocol,
             Section 3.5.1.";
```





```
        }

        leaf selected {
          type boolean;
          description
            "A boolean flag indicating whether this route is selected,
             i.e., whether it is currently being used for forwarding and
             is being advertised.";
          reference
            "RFC YYYY, Babel Information Model, Section 3.6.";
        }
        description
```

```
          "A set of babel-route-obj objects. Includes received and
           routes routes.";
        reference
          "RFC YYYY, Babel Information Model, Section 3.1.";
      }

      list security {
        key "mechanism";

        uses security;
        description
          "A security-obj object that applies to all interfaces. If this
           object is implemented, it allows a security mechanism to be
           enabled or disabled in a manner that applies to all Babel
           messages on all interfaces";
        reference
          "RFC YYYY, Babel Information Model, Section 3.1.";
      }
    }
  }
```

<CODE ENDS>

# 3. IANA Considerations

This document registers ?? URIs and ?? YANG modules.

## 3.1. URI Registrations

## 3.2. YANG Module Name Registration

This document registers ?? YANG module in the YANG Module Names registry YANG [RFC6020].





```
name:
namespace: urn:ietf:params:xml:ns:yang:
prefix: babel
reference: RFC XXXX
```

## 4. Security Considerations

The YANG module specified in this document defines a schema for data that is designed to be accessed via network management protocol such as NETCONF [RFC6241] or RESTCONF [RFC8040]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The lowest RESTCONF layer is HTTPS, and the mandatory-to-implement secure transport is TLS [RFC8446].

The NETCONF Access Control Model (NACM [RFC8341]) provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

These are the subtrees and data nodes and their sensitivity/ vulnerability:

## 5. Acknowledgements

## 6. References

### 6.1. Normative References


[I-D.ietf-babel-rfc6126bis]
Chroboczek, J. and D. Schinazi, "The Babel Routing Protocol", draft-ietf-babel-rfc6126bis-05 (work in progress), May 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>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[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>.

[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>.

## 6.2. Informative References

[I-D.ietf-babel-information-model] Stark, B., "Babel Information Model", draft-ietf-babel-information-model-03 (work in progress), June 2018.

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

[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>.

[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>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

## Appendix A. An Appendix

Authors' Addresses

Mahesh Jethanandani
VMware
California
USA

Email: mjethanandani@gmail.com

Barbara Stark
AT&T
Atlanta, GA
USA

Email: barbara.stark@att.com