ISIS Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 5, 2015

S. Litkowski
Orange
D. Yeung
A. Lindem
Cisco Systems
J. Zhang
Juniper Networks
L. Lhotka
March 04, 2015

YANG Data Model for ISIS protocol
draft-ietf-isis-yang-isis-cfg-02

Abstract

This document defines a YANG data model that can be used to configure and manage ISIS protocol on network elements.

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 [RFC2119].

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 September 5, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
1.1. Tree diagram 2
2. Design of the Data Model 3
2.1. ISIS Configuration 5
2.2. Multitopology Parameters 7
2.3. Per-Level Parameters 7
2.4. Per-Interface Parameters 8
2.5. ISO parameters 10
2.6. IP FRR 10
2.7. Segment Routing 10
2.8. Operational State 11
3. RPC Operations 18
4. Notifications 19
5. Interaction with Other YANG Modules 23
6. YANG Module 23
7. Security Considerations 110
8. Contributors 111
9. Acknowledgements 112
10. IANA Considerations 112
11. Normative References 112
Appendix A. Example: NETCONF <get> Reply 112
Authors' Addresses 115

1. Introduction

This document defines a YANG data model for ISIS routing protocol.

The data model covers configuration of an ISIS routing protocol instance as well as operational states.

1.1. Tree diagram

A simplified graphical representation of the data model is presented in Section 2.

The meaning of the symbols in these diagrams is as follows:

- o Brackets "[" and "]" enclose list keys.

- o Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.

- o Abbreviations before data node names: "rw" means configuration (read-write), and "ro" state data (read-only).

- o Symbols after data node names: "?" means an optional node and "*" denotes a "list" or "leaf-list".

- o Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- o Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Design of the Data Model

The ISIS YANG module is divided in two main "isis" containers that are augmenting the "routing-protocol" lists in ietf-routing module with specific ISIS parameters.

One container contains the writable parameters, while the other contains the operational states.

The figure below describe the overall structure of the isis YANG module:

```
module: ietf-isis
augment /rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route:
   +--ro metric?       uint32
   +--ro tag*          uint64
   +--ro route-type?   enumeration
   +--ro segment-id?   uint32 {segment-routing}?
augment /rt:active-route/rt:output/rt:route:
   +--ro metric?       uint32
   +--ro tag*          uint64
   +--ro route-type?   enumeration
   +--ro segment-id?   uint32 {segment-routing}?
augment /if:interfaces/if:interface:
   +--rw clns-mtu?   uint16
augment /rt:routing/rt:routing-instance/rt:routing-protocols/rt:routing-
protocol:
   +--rw isis
      +--rw instance* [routing-instance]
         +--rw routing-instance          rt:routing-instance-ref
         +--rw level-type?               level
         +--rw system-id?                system-id
```

```
         +--rw maximum-area-addresses?   uint8 {maximum-area-addresses}?
         +--rw area-address*             area-address
         +--rw mpls
         |     ...
         +--rw reference-bandwidth?      uint32 {reference-bandwidth}?
         +--rw lsp-mtu?                  uint16
         +--rw lsp-lifetime?             uint16
         +--rw lsp-refresh?              uint16 {lsp-refresh}?
         +--rw graceful-restart {graceful-restart}?
         |     ...
         +--rw node-tag {node-tag}?
         |     ...
         +--rw fast-reroute {fast-reroute}?
         |     ...
         +--rw segment-routing {segment-routing}?
         |     ...
         +--rw authentication* [level]
         |     ...
         +--rw metric-type* [level]
         |     ...
         +--rw preference* [level]
         |     ...
         +--rw default-metric* [level]
         |     ...
         +--rw af* [af] {nlpid-control}?
         |     ...
         +--rw topologies* [name] {multi-topology}?
         |     ...
         +--rw overload* [level]
         |     ...
         +--rw overload-max-metric* [level] {overload-max-metric}?
         |     ...
         +--rw interfaces
               ...
augment /rt:routing-state/rt:routing-instance/rt:routing-protocols/rt:routing-
protocol:
   +--ro isis
      +--ro instance* [routing-instance]
         +--ro routing-instance    rt:routing-instance-ref
         +--ro fast-reroute {fast-reroute}?
         |     ...
         +--ro topologies* [name]
         |     ...
         +--ro system-counters
         |     ...
         +--ro interfaces
         |     ...
         +--ro spf-log
```

```
          |     ...
```

```
         +--ro lsp-log
         |     ...
         +--ro database
         |     ...
         +--ro hostnames
               ...
```

2.1. ISIS Configuration

The ISIS configuration currently supports both VRF-centric and protocol-centric configuration. This may be changed in future.

In a protocol-centric configuration, the isis configuration is applied within the standard-routing-instance and the instance list helps to reference the routing-instance where ISIS is activated.

In a VRF-centric configuration, the isis configuration is applied directly within the appropriate routing-instance where ISIS is activated. In this case, the instance list will contain a single element.

The ISIS configuration container is divided in:

o Global parameters.

o Per interface configuration (see Section 2.4).

It would to up to extension modules to augment this model to support vendor specific parameters.

```
augment /rt:routing/rt:routing-instance/rt:routing-protocols/rt:routing-
protocol:
   +--rw isis
      +--rw instance* [routing-instance]
         +--rw routing-instance          rt:routing-instance-ref
         +--rw level-type?               level
         +--rw system-id?                system-id
         +--rw maximum-area-addresses?   uint8 {maximum-area-addresses}?
         +--rw area-address*             area-address
         +--rw mpls
         |  +--rw ipv4-router-id?   inet:ipv4-address {ipv4-router-id}?
         |  +--rw ipv6-router-id?   inet:ipv6-address {ipv6-router-id}?
         |  +--rw igp-ldp-sync {igp-ldp-sync}?
         +--rw reference-bandwidth?      uint32 {reference-bandwidth}?
         +--rw lsp-mtu?                  uint16
         +--rw lsp-lifetime?             uint16
         +--rw lsp-refresh?              uint16 {lsp-refresh}?
         +--rw graceful-restart {graceful-restart}?
```

```
          |  +--rw enabled?   boolean
          +--rw node-tag {node-tag}?
          |  +--rw node-tag* [tag]
          |     +--rw tag    uint32
          +--rw fast-reroute {fast-reroute}?
          |  +--rw lfa {lfa}?
          +--rw segment-routing {segment-routing}?
          |  +--rw enabled?    boolean
          |  +--rw bindings
          |     +--rw advertise?   boolean
          |     +--rw receive?     boolean
          +--rw authentication* [level]
          |  +--rw (authentication-type)?
          |  |  +--:(key-chain) {key-chain}?
          |  |  |  +--rw key-chain?      key-chain:key-chain-ref
          |  |  +--:(password)
          |  |     +--rw key?            string
          |  |     +--rw (algorithm)?
          |  |        +--:(hmac-sha1-12)
          |  |        |  +--rw hmac-sha1-12?   empty
          |  |        +--:(hmac-sha1-20)
          |  |        |  +--rw hmac-sha1-20?   empty
          |  |        +--:(md5)
          |  |        |  +--rw md5?            empty
          |  |        +--:(sha-1)
          |  |        |  +--rw sha-1?          empty
          |  |        +--:(hmac-sha-1)
          |  |        |  +--rw hmac-sha-1?     empty
          |  |        +--:(hmac-sha-256)
          |  |        |  +--rw hmac-sha-256?   empty
          |  |        +--:(hmac-sha-384)
          |  |        |  +--rw hmac-sha-384?   empty
          |  |        +--:(hmac-sha-512)
          |  |           +--rw hmac-sha-512?   empty
          |  +--rw level           level
          +--rw metric-type* [level]
          |  +--rw value?   enumeration
          |  +--rw level    level
          +--rw preference* [level]
          |  +--rw (granularity)?
          |  |  +--:(detail)
          |  |  |  +--rw internal?   uint8
          |  |  |  +--rw external?   uint8
          |  |  +--:(coarse)
          |  |     +--rw default?    uint8
          |  +--rw level       level
          +--rw default-metric* [level]
          |  +--rw value?   wide-metric
```

```
         |  +--rw level    level
         +--rw af* [af] {nlpid-control}?
         |  +--rw af         identityref
         |  +--rw enabled?   boolean
         +--rw topologies* [name] {multi-topology}?
         |  +--rw enabled?           boolean
         |  +--rw name               rt:rib-ref
         |  +--rw fast-reroute {fast-reroute}?
         |  |  +--rw lfa {lfa}?
         |  +--rw segment-routing {segment-routing}?
         |  |  +--rw enabled?    boolean
         |  |  +--rw bindings
         |  |     +--rw advertise?   boolean
         |  |     +--rw receive?     boolean
         |  +--rw default-metric* [level]
         |  |  +--rw value?   wide-metric
         |  |  +--rw level    level
         |  +--rw node-tag {node-tag}?
         |     +--rw node-tag* [tag]
         |        +--rw tag    uint32
         +--rw overload* [level]
         |  +--rw status?    boolean
         |  +--rw timeout?   uint16
         |  +--rw level      level
         +--rw overload-max-metric* [level] {overload-max-metric}?
         |  +--rw status?    boolean
         |  +--rw timeout?   uint16
         |  +--rw level      level
         +--rw interfaces
            +--rw interface* [name]
                        ...
```

2.2. Multitopology Parameters

The "topologies" list is used to enable support of MT extensions for specific address families.

Each topology should refer to an existing RIB.

Some specific parameters could be defined for a specific topology at global level and also at interface level.

2.3. Per-Level Parameters

Some parameters support per level configuration. In this case, the parameter is built as a list, so different values could be used for each level. The "level-all" permits to apply a value to both levels.

```
          +--rw priority* [level]
          |  +--rw value?   uint8
          |  +--rw level    level
```

Example :

```
          <priority>
              <value>100</value>
                          <level>level-1</level>
          </priority>
                  <priority>
              <value>200</value>
                          <level>level-2</level>
          </priority>
```

2.4. Per-Interface Parameters

The per-interface section of the ISIS instance describes the interface specific parameters.

The interface is a reference to an interface in the Interface YANG model.

Each interface has interface-specific parameters that may have a different value per level as described in previous section. An interface-specific parameter always override an ISIS global parameter .

Some parameters like BFD and hello-padding are defined as containers to permit easy extension by vendor specific modules.

```
 +--rw interfaces
        +--rw interface* [name]
           +--rw name                       if:interface-ref
           +--rw level-type?                level
           +--rw lsp-pacing-interval?       uint16
           +--rw lsp-retransmit-interval?   uint16
           +--rw passive?                   boolean
           +--rw csnp-interval?             uint16
           +--rw hello-padding
           |  +--rw enabled?   boolean
           +--rw mesh-group-enabled?        mesh-group-state
           +--rw mesh-group?                uint8
           +--rw interface-type?            interface-type
           +--rw enabled?                   boolean
           +--rw tag*                       uint32 {prefix-tag}?
           +--rw tag64*                     uint64 {prefix-tag64}?
           +--rw hello-authentication* [level]
```

```
               |  +--rw (authentication-type)?
               |  |  +--:(key-chain) {key-chain}?
               |  |  |  +--rw key-chain?      key-chain:key-chain-ref
               |  |  +--:(password)
               |  |     +--rw key?            string
               |  |     +--rw (algorithm)?
               |  |           ...
               |  +--rw level           level
               +--rw hello-interval* [level]
               |  +--rw value?   uint16
               |  +--rw level    level
               +--rw hello-multiplier* [level]
               |  +--rw value?   uint16
               |  +--rw level    level
               +--rw priority* [level]
               |  +--rw value?   uint8
               |  +--rw level    level
               +--rw metric* [level]
               |  +--rw value?   wide-metric
               |  +--rw level    level
               +--rw af* [af]
               |  +--rw af                 identityref
               |  +--rw segment-routing {segment-routing}?
               |  |  +--rw prefix-sid* [value]
               |  |     +--rw value-type?          enumeration
               |  |     +--rw value                uint32
               |  |     +--rw node-flag?           boolean
               |  |     +--rw last-hop-behavior?   enumeration
               |  +--rw bfd {bfd}?
               |     +--rw enabled?   boolean
               +--rw topologies* [name]
               |  +--rw name               rt:rib-ref
               |  +--rw metric* [level]
               |  |  +--rw value?   wide-metric
               |  |  +--rw level    level
               |  +--rw fast-reroute {fast-reroute}?
               |  |  +--rw lfa* [level] {lfa}?
               |  |     +--rw candidate-disabled?   boolean
               |  |     +--rw enabled?              boolean
               |  |     +--rw remote-lfa {remote-lfa}?
               |  |     |     ...
               |  |     +--rw level                 level
               |  +--rw segment-routing {segment-routing}?
               |     +--rw authorize-bundle
               |     |  +--rw enabled?   boolean
               |     |  +--rw weight?    uint8
               |     +--rw advertise-protection?   enumeration
               +--rw fast-reroute {fast-reroute}?
```

```
             |  +--rw lfa* [level] {lfa}?
             |     +--rw candidate-disabled?   boolean
             |     +--rw enabled?              boolean
             |     +--rw remote-lfa {remote-lfa}?
             |     |  +--rw enabled?   boolean
             |     +--rw level                 level
             +--rw segment-routing {segment-routing}?
             |  +--rw authorize-bundle
             |  |  +--rw enabled?   boolean
             |  |  +--rw weight?    uint8
             |  +--rw advertise-protection?   enumeration
             +--rw mpls
                    +--rw igp-ldp-sync {igp-ldp-sync}?
                           +--rw enabled?   boolean
```

2.5. ISO parameters

Some ISO parameters may be required.

This module augments interface configuration model to support ISO configuration parameters.

The clns-mtu can be defined under the interface.

2.6. IP FRR

This YANG model supports LFA and remote LFA as IP FRR techniques. The "fast-reroute" container may be augmented by other models to support other IPFRR flavors (MRT ...).

The current version of the model supports activation of LFA and remote LFA at interface only. The global "lfa" container is present but kept empty to permit augmentation with vendor specific properties like policies.

Remote LFA is considered as a child of LFA. Remote LFA cannot be enabled if LFA is not enabled.

The "candidate-disabled" permit to mark an interface to not be used as a backup.

2.7. Segment Routing

This item is still under discussion and some part of the proposed configuration may fit in a segment-routing specific model.

2.8. Operational State

"isis" container provides operational states for ISIS. This container is divided in multiple components:

- o system-counters : provides statistical informations about the global system.

- o interface : provides configuration state information for each interface.

- o adjacencies: provides state information about current ISIS adjacencies.

- o spf-log: provides information about SPF events on the node.

- o lsp-log: provides information about LSP events on the node (reception of an LSP or modification of local LSP).

- o database: provides details on current LSDB.

- o hostnames: provides information about system-id to hostname mappings.

- o fast-reroute: provides information about IP FRR.

- o segment-routing: provides information about segment-routing.

```
augment /rt:routing-state/rt:routing-instance/rt:routing-protocols/rt:routing-
protocol:
   +--ro isis
      +--ro instance* [routing-instance]
         +--ro routing-instance    rt:routing-instance-ref
         +--ro fast-reroute {fast-reroute}?
         |  +--ro protected-routes
         |  |  +--ro af-stats* [af prefix alternate]
         |  |     +--ro af                      identityref
         |  |     +--ro prefix                  string
         |  |     +--ro alternate               string
         |  |     +--ro alternate-type?         enumeration
         |  |     +--ro best?                   boolean
         |  |     +--ro non-best-reason?        string
         |  |     +--ro protection-available?   bits
         |  |     +--ro alternate-metric1?      uint32
         |  |     +--ro alternate-metric2?      uint32
         |  |     +--ro alternate-metric3?      uint32
         |  +--ro nonprotected-routes
         |  |  +--ro af-stats* [af prefix]
         |  |     +--ro af        identityref
```

```
         |  |     +--ro prefix    string
         |  +--ro protection-statistics* [frr-protection-method]
         |     +--ro frr-protection-method    string
         |     +--ro af-stats* [af]
         |        +--ro af                      identityref
         |        +--ro total-routes?           uint32
         |        +--ro unprotected-routes?     uint32
         |        +--ro protected-routes?       uint32
         |        +--ro linkprotected-routes?   uint32
         |        +--ro nodeprotected-routes?   uint32
         +--ro topologies* [name]
         |  +--ro name          rt:rib-ref
         |  +--ro fast-route {fast-reroute}?
         |     +--ro protected-routes
         |     |  +--ro af-stats* [af prefix alternate]
         |     |     +--ro af                      identityref
         |     |     +--ro prefix                  string
         |     |     +--ro alternate               string
         |     |     +--ro alternate-type?         enumeration
         |     |     +--ro best?                   boolean
         |     |     +--ro non-best-reason?        string
         |     |     +--ro protection-available?   bits
         |     |     +--ro alternate-metric1?      uint32
         |     |     +--ro alternate-metric2?      uint32
         |     |     +--ro alternate-metric3?      uint32
         |     +--ro nonprotected-routes
         |     |  +--ro af-stats* [af prefix]
         |     |     +--ro af        identityref
         |     |     +--ro prefix    string
         |     +--ro protection-statistics* [frr-protection-method]
         |        +--ro frr-protection-method    string
         |        +--ro af-stats* [af]
         |           +--ro af                      identityref
         |           +--ro total-routes?           uint32
         |           +--ro unprotected-routes?     uint32
         |           +--ro protected-routes?       uint32
         |           +--ro linkprotected-routes?   uint32
         |           +--ro nodeprotected-routes?   uint32
         +--ro system-counters
         |  +--ro level* [level]
         |     +--ro level                            level-number
         |     +--ro corrupted-lsps?                  uint32
         |     +--ro authentication-type-fails?       uint32
         |     +--ro authentication-fails?            uint32
         |     +--ro database-overload?               uint32
         |     +--ro own-lsp-purge?                   uint32
         |     +--ro manual-address-drop-from-area?   uint32
         |     +--ro max-sequence?                    uint32
```

```
          |     +--ro sequence-number-skipped?         uint32
          |     +--ro id-len-mismatch?                 uint32
          |     +--ro partition-changes?               uint32
          |     +--ro lsp-errors?                      uint32
          |     +--ro spf-runs?                        uint32
          +--ro interfaces
          |  +--ro interfaces* [interface]
          |     +--ro interface          string
          |     +--ro status
          |     |  +--ro circuit-id?            circuit-id
          |     |  +--ro admin-state?           admin-state
          |     |  +--ro interface-type?        interface-type
          |     |  +--ro passive?               empty
          |     |  +--ro three-way-handshake?   boolean
          |     |  +--ro mesh-group-enabled?    mesh-group-state
          |     |  +--ro mesh-group?            uint8
          |     |  +--ro level* [level]
          |     |     +--ro level                level-number
          |     |     +--ro priority?            uint8
          |     |     +--ro hello-multiplier?    uint16
          |     |     +--ro hello-interval?      uint16
          |     |     +--ro dr-hello-interval?   uint32
          |     |     +--ro metric?              wide-metric
          |     +--ro fast-reroute {fast-reroute}?
          |     |  +--ro lfa* [level] {lfa}?
          |     |     +--ro candidate-disabled?   boolean
          |     |     +--ro enabled?              boolean
          |     |     +--ro remote-lfa {remote-lfa}?
          |     |     |  +--ro enabled?   boolean
          |     |     +--ro level                 level-number
          |     +--ro adjacencies
          |     |  +--ro adjacency*
          |     |     +--ro neighbor-systype?               level
          |     |     +--ro neighbor-sysid?                 system-id
          |     |     +--ro neighbor-extended-circuit-id?   extended-circuit-id
          |     |     +--ro neighbor-snpa?                  snpa
          |     |     +--ro usage?                          level
          |     |     +--ro hold-timer?                     uint16
          |     |     +--ro neighbor-priority?              uint8
          |     |     +--ro lastuptime?                     yang:timestamp
          |     |     +--ro state?                          enumeration
          |     |     +--ro adjacency-sid* [value] {segment-routing}?
          |     |        +--ro af?                     identityref
          |     |        +--ro value                   uint32
          |     |        +--ro weight?                 uint8
          |     |        +--ro protection-requested?   boolean
          |     +--ro topologies* [name]
          |     |  +--ro name            rt:rib-ref
```

```
          |     |  +--ro fast-reroute {fast-reroute}?
          |     |  |  +--ro lfa* [level] {lfa}?
          |     |  |     +--ro candidate-disabled?   boolean
          |     |  |     +--ro enabled?              boolean
          |     |  |     +--ro remote-lfa {remote-lfa}?
          |     |  |     |     ...
          |     |  |     +--ro level                 level-number
          |     |  +--ro adjacencies
          |     |  |  +--ro adjacency*
          |     |  |     +--ro neighbor-systype?               level
          |     |  |     +--ro neighbor-sysid?                 system-id
          |     |  |     +--ro neighbor-extended-circuit-id?   extended-circuit-
id
          |     |  |     +--ro neighbor-snpa?                  snpa
          |     |  |     +--ro usage?                          level
          |     |  |     +--ro hold-timer?                     uint16
          |     |  |     +--ro neighbor-priority?              uint8
          |     |  |     +--ro lastuptime?                     yang:timestamp
          |     |  |     +--ro state?                          enumeration
          |     |  |     +--ro adjacency-sid* [value] {segment-routing}?
          |     |  |           ...
          |     |  +--ro level* [level]
          |     |     +--ro level     level-number
          |     |     +--ro metric?   wide-metric
          |     +--ro event-counters
          |     |  +--ro adjacency-changes?             uint32
          |     |  +--ro adjacency-number?              uint32
          |     |  +--ro init-fails?                    uint32
          |     |  +--ro adjacency-rejects?             uint32
          |     |  +--ro id-len-mismatch?               uint32
          |     |  +--ro max-area-addresses-mismatch?   uint32
          |     |  +--ro authentication-type-fails?     uint32
          |     |  +--ro authentication-fails?          uint32
          |     |  +--ro lan-dis-changes?               uint32
          |     +--ro packet-counters
          |        +--ro level* [level]
          |           +--ro level      level-number
          |           +--ro iih
          |           |  +--ro in?    uint32
          |           |  +--ro out?   uint32
          |           +--ro ish
          |           |  +--ro in?    uint32
          |           |  +--ro out?   uint32
          |           +--ro esh
          |           |  +--ro in?    uint32
          |           |  +--ro out?   uint32
          |           +--ro lsp
          |           |  +--ro in?    uint32
```

```
          |           |  +--ro out?   uint32
```

```
         |           +--ro psnp
         |           |  +--ro in?    uint32
         |           |  +--ro out?   uint32
         |           +--ro csnp
         |           |  +--ro in?    uint32
         |           |  +--ro out?   uint32
         |           +--ro unknown
         |              +--ro in?    uint32
         |              +--ro out?   uint32
         +--ro spf-log
         |  +--ro event* [id]
         |     +--ro id                    uint32
         |     +--ro spf-type?             enumeration
         |     +--ro level?                level-number
         |     +--ro spf-delay?            uint32
         |     +--ro schedule-timestamp?   yang:timestamp
         |     +--ro start-timestamp?      yang:timestamp
         |     +--ro end-timestamp?        yang:timestamp
         |     +--ro trigger-lsp* [lsp]
         |        +--ro lsp         lsp-id
         |        +--ro sequence?   uint32
         +--ro lsp-log
         |  +--ro event* [id]
         |     +--ro id                    uint32
         |     +--ro level?                level-number
         |     +--ro lsp
         |     |  +--ro lsp?        lsp-id
         |     |  +--ro sequence?   uint32
         |     +--ro received-timestamp?   yang:timestamp
         |     +--ro change?               bits
         +--ro database
         |  +--ro level-db* [level]
         |     +--ro level    level-number
         |     +--ro lsp* [lsp-id]
         |        +--ro lsp-id                          lsp-id
         |        +--ro checksum?                       uint16
         |        +--ro remaining-lifetime?             uint16
         |        +--ro sequence?                       uint32
         |        +--ro attributes?                     bits
         |        +--ro is-neighbor
         |        |  +--ro neighbor* [neighbor-id]
         |        |     +--ro neighbor-id       system-id
         |        |     +--ro i-e?              boolean
         |        |     +--ro default-metric?   std-metric
         |        |     +--ro delay-metric
         |        |     |     ...
         |        |     +--ro expense-metric
         |        |     |     ...
```

```
          |        |     +--ro error-metric
          |        |           ...
          |        +--ro authentication
          |        |  +--ro authentication-type?   string
          |        |  +--ro authentication-key?    string
          |        +--ro extended-is-neighbor
          |        |  +--ro neighbor* [neighbor-id]
          |        |     +--ro neighbor-id             system-id
          |        |     +--ro metric?                 wide-metric
          |        |     +--ro adjacency-segment-id* [value]
          |        |           ...
          |        +--ro ipv4-internal-reachability
          |        |  +--ro prefixes* [ip-prefix]
          |        |     +--ro up-down?          boolean
          |        |     +--ro i-e?              boolean
          |        |     +--ro ip-prefix         inet:ipv4-address
          |        |     +--ro prefix-len?       uint8
          |        |     +--ro default-metric?   std-metric
          |        |     +--ro delay-metric
          |        |     |     ...
          |        |     +--ro expense-metric
          |        |     |     ...
          |        |     +--ro error-metric
          |        |           ...
          |        +--ro protocol-supported*              uint8
          |        +--ro ipv4-external-reachability
          |        |  +--ro prefixes* [ip-prefix]
          |        |     +--ro up-down?          boolean
          |        |     +--ro i-e?              boolean
          |        |     +--ro ip-prefix         inet:ipv4-address
          |        |     +--ro prefix-len?       uint8
          |        |     +--ro default-metric?   std-metric
          |        |     +--ro delay-metric
          |        |     |     ...
          |        |     +--ro expense-metric
          |        |     |     ...
          |        |     +--ro error-metric
          |        |           ...
          |        +--ro ipv4-addresses*                  inet:ipv4-address
          |        +--ro ipv4-te-routerid?                inet:ipv4-address
          |        +--ro extended-ipv4-reachability
          |        |  +--ro prefixes* [ip-prefix]
          |        |     +--ro up-down?             boolean
          |        |     +--ro ip-prefix            inet:ipv4-address
          |        |     +--ro prefix-len?          uint8
          |        |     +--ro metric?              wide-metric
          |        |     +--ro tag*                 uint32
          |        |     +--ro tag64*               uint64
```

```
          |        |     +--ro prefix-segment-id* [value]
          |        |           ...
          |        +--ro dynamic-hostname?               string
          |        +--ro ipv6-te-routerid?               inet:ipv6-address
          |        +--ro mt-is-neighbor
          |        |  +--ro neighbor* [neighbor-id]
          |        |     +--ro MT-ID?                 uint16
          |        |     +--ro neighbor-id            system-id
          |        |     +--ro metric?                wide-metric
          |        |     +--ro adjacency-segment-id* [value]
          |        |           ...
          |        +--ro mt-entries
          |        |  +--ro topology* [MT-ID]
          |        |     +--ro MT-ID         uint16
          |        |     +--ro attributes?   bits
          |        +--ro ipv6-addresses*                 inet:ipv6-address
          |        +--ro mt-extended-ipv4-reachability
          |        |  +--ro prefixes* [ip-prefix]
          |        |     +--ro MT-ID?              uint16
          |        |     +--ro up-down?            boolean
          |        |     +--ro ip-prefix           inet:ipv4-address
          |        |     +--ro prefix-len?         uint8
          |        |     +--ro metric?             wide-metric
          |        |     +--ro tag*                uint32
          |        |     +--ro tag64*              uint64
          |        |     +--ro prefix-segment-id* [value]
          |        |           ...
          |        +--ro mt-ipv6-reachability
          |        |  +--ro prefixes* [ip-prefix]
          |        |     +--ro MT-ID?              uint16
          |        |     +--ro up-down?            boolean
          |        |     +--ro ip-prefix           inet:ipv6-address
          |        |     +--ro prefix-len?         uint8
          |        |     +--ro metric?             wide-metric
          |        |     +--ro tag*                uint32
          |        |     +--ro tag64*              uint64
          |        |     +--ro prefix-segment-id* [value]
          |        |           ...
          |        +--ro ipv6-reachability
          |        |  +--ro prefixes* [ip-prefix]
          |        |     +--ro up-down?            boolean
          |        |     +--ro ip-prefix           inet:ipv6-address
          |        |     +--ro prefix-len?         uint8
          |        |     +--ro metric?             wide-metric
          |        |     +--ro tag*                uint32
          |        |     +--ro tag64*              uint64
          |        |     +--ro prefix-segment-id* [value]
          |        |           ...
```

```
         |        +--ro segment-routing-bindings* [fec range] {segment-
routing}?
         |        |  +--ro fec        string
         |        |  +--ro range      uint16
         |        |  +--ro flags?     bits
         |        |  +--ro weight?    uint8
         |        |  +--ro binding*
         |        |     +--ro prefix-sid
         |        |     |     ...
         |        |     +--ro ero-metric?                          uint32
         |        |     +--ro ero
         |        |     |     ...
         |        |     +--ro backup-ero
         |        |     |     ...
         |        |     +--ro unnumbered-interface-id-ero
         |        |     |     ...
         |        |     +--ro backup-unnumbered-interface-id-ero
         |        |           ...
         |        +--ro router-capabilities*
         |           +--ro flags?                       bits
         |           +--ro node-tag {node-tag}?
         |           |  +--ro node-tag*
         |           |        ...
         |           +--ro segment-routing*
         |           |  +--ro flags?            bits
         |           |  +--ro range?            uint32
         |           |  +--ro starting-value?   uint32
         |           +--ro segment-routing-algorithm*   uint8
         |           +--ro binding?                     binary
         +--ro hostnames
            +--ro hostname* [system-id]
               +--ro system-id    system-id
               +--ro hostname?    string
```

3. RPC Operations

The "ietf-isis" module defines two RPC operations:

- o clear-isis-database: reset the content of a particular ISIS database and restart database synchronization with the neighbors.

- o clear-isis-adjacency: restart a particular set of ISIS adjacencies.

```
rpcs:
   +---x clear-adjacency
   |  +--ro input
   |     +--ro routing-instance-name             rt:routing-instance-state-ref
   |     +--ro routing-protocol-instance-name    instance-state-ref
   |     +--ro level?                            level
   |     +--ro interface?                        string
   +---x clear-database
      +--ro input
         +--ro routing-instance-name             rt:routing-instance-state-ref
         +--ro routing-protocol-instance-name    instance-state-ref
         +--ro level?                            level
```

4. Notifications

The "ietf-isis" module introduces some notifications :

database-overload : raised when overload condition is changed.

lsp-too-large : raised when the system tries to propagate a too large PDU.

corrupted-lsp-detected : raised when the system find that an LSP that was stored in memory has become corrupted.

attempt-to-exceed-max-sequence : This notification is sent when the system wraps the 32-bit sequence counter of an LSP.

id-len-mismatch : This notification is sent when we receive a PDU with a different value for the System ID length.

max-area-addresses-mismatch : This notification is sent when we receive a PDU with a different value for the Maximum Area Addresses.

own-lsp-purge : This notification is sent when the system receives a PDU with its own system ID and zero age.

sequence-number-skipped : This notification is sent when the system receives a PDU with its own system ID and different contents. The system has to reissue the LSP with a higher sequence number.

authentication-type-failure : This notification is sent when the system receives a PDU with the wrong authentication type field.

authentication-failure : This notification is sent when the system receives a PDU with the wrong authentication information.

version-skew : This notification is sent when the system receives a PDU with a different protocol version number.

area-mismatch : This notification is sent when the system receives a Hello PDU from an IS that does not share any area address.

rejected-adjacency : This notification is sent when the system receives a Hello PDU from an IS but does not establish an adjacency for some reason.

protocols-supported-mismatch : This notification is sent when the system receives a non pseudonode LSP that has no matching protocol supported.

lsp-error-detected : This notification is sent when the system receives a LSP with a parse error.

adjacency-change : This notification is sent when an ISIS adjacency moves to Up state or to Down state.

lsp-received : This notification is sent when a LSP is received.

lsp-generation : This notification is sent when a LSP is regenerated.

```
notifications:
   +---n database-overload
   |  +--ro instance-name?    string
   |  +--ro instance-level?   level
   |  +--ro overload?         enumeration
   +---n lsp-too-large
   |  +--ro instance-name?         string
   |  +--ro instance-level?        level
   |  +--ro interface-name?        string
   |  +--ro interface-level?       level
   |  +--ro extended-circuit-id?   extended-circuit-id
   |  +--ro pdu-size?              uint32
   |  +--ro lsp-id?                lsp-id
   +---n corrupted-lsp-detected
   |  +--ro instance-name?    string
   |  +--ro instance-level?   level
   |  +--ro lsp-id?           lsp-id
   +---n attempt-to-exceed-max-sequence
   |  +--ro instance-name?    string
   |  +--ro instance-level?   level
   |  +--ro lsp-id?           lsp-id
   +---n id-len-mismatch
   |  +--ro instance-name?         string
```

```
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro pdu-field-len?         uint8
      |  +--ro raw-pdu?               binary
      +---n max-area-addresses-mismatch
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro max-area-addresses?    uint8
      |  +--ro raw-pdu?               binary
      +---n own-lsp-purge
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro lsp-id?                lsp-id
      +---n sequence-number-skipped
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro lsp-id?                lsp-id
      +---n authentication-type-failure
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro raw-pdu?               binary
      +---n authentication-failure
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro raw-pdu?               binary
      +---n version-skew
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
```

```
      |  +--ro protocol-version?      uint8
      |  +--ro raw-pdu?               binary
      +---n area-mismatch
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro raw-pdu?               binary
      +---n rejected-adjacency
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro raw-pdu?               binary
      |  +--ro reason?                string
      +---n protocols-supported-mismatch
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro raw-pdu?               binary
      |  +--ro protocols*             uint8
      +---n lsp-error-detected
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro lsp-id?                lsp-id
      |  +--ro raw-pdu?               binary
      |  +--ro error-offset?          uint32
      |  +--ro tlv-type?              uint8
      +---n adjacency-change
      |  +--ro instance-name?         string
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro neighbor?              string
      |  +--ro neighbor-system-id?    system-id
      |  +--ro level?                 level
      |  +--ro state?                 enumeration
      |  +--ro reason?                string
      +---n lsp-received
      |  +--ro instance-name?         string
```

```
      |  +--ro instance-level?        level
      |  +--ro interface-name?        string
      |  +--ro interface-level?       level
      |  +--ro extended-circuit-id?   extended-circuit-id
      |  +--ro lsp-id?                lsp-id
      |  +--ro sequence?              uint32
      |  +--ro received-timestamp?    yang:timestamp
      |  +--ro neighbor-system-id?    system-id
      +---n lsp-generation
         +--ro instance-name?    string
         +--ro instance-level?   level
         +--ro lsp-id?           lsp-id
         +--ro sequence?         uint32
         +--ro send-timestamp?   yang:timestamp
```

5. Interaction with Other YANG Modules

The "isis" configuration container augments the "/rt:routing/rt:routing-instance/rt:routing-protocols/routing-protocol" container of the ietf-routing module by defining ISIS specific parameters.

The "isis" module augments "/if:interfaces/if:interface" with ISO specific parameters.

The "isis" operational state container augments the "/rt:routing-state/rt:routing-instance/rt:routing-protocols/routing-protocol" container of the ietf-routing module by defining ISIS specific operational states.

Some ISIS specific routes attributes are added to route objects of the ietf-routing module by augmenting "/rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route" and "/rt:active-route/rt:output/rt:route".

6. YANG Module

```
<CODE BEGINS> file "ietf-isis@2015-03-04.yang"

module ietf-isis {
  namespace "urn:ietf:params:xml:ns:yang:ietf-isis";

  prefix isis;

  import ietf-routing {
            prefix "rt";
        }

  import ietf-inet-types {
```

```
       prefix inet;
     }

     import ietf-yang-types {
       prefix yang;
     }

     import ietf-interfaces {
       prefix "if";
     }

     organization
      "IETF ISIS Working Group";

     contact
       "WG List:  &lt;mailto:isis-wg@ietf.org&gt;

       Editor:    Stephane Litkowski
             &lt;mailto:stephane.litkowski@orange.com&gt;

           Derek Yeung
             &lt;mailto:myeung@cisco.com&gt;
           Acee Lindem
             &lt;mailto:acee@cisco.com&gt;
           Jeffrey Zhang
             &lt;mailto:zzhang@juniper.net&gt;
           Ladislav Lhotka
             &lt;mailto:llhotka@nic.cz&gt;
           Yi Yang
             &lt;mailto:yiya@cisco.com&gt;
           Dean Bogdanovic
             &lt;mailto:deanb@juniper.net&gt;
           Kiran Agrahara Sreenivasa
             &lt;mailto:kkoushik@brocade.com&gt;
           Yingzhen Qu
             &lt;mailto:yiqu@cisco.com&gt;

       ";

     description
       "The YANG module defines a generic configuration model for
       ISIS common across all of the vendor implementations.";

     revision 2015-03-03 {
       description
         "
           * Defining hierarchy for operational states
           * Adding CLNS MTU
```

```
        ";
      reference "draft-ietf-isis-yang-isis-02";
    }
    revision 2015-02-20 {
      description
        "
          * Removing igp-ldp-sync timer in IS-IS

        ";
      reference "";
    }
    revision 2014-12-15 {
      description
        "
          * Adding IPFRR
          * Adding igp-ldp sync
          * Adding segment routing
          * Adding instance reference to operational states.
          * Move AF type from string to identity
          * Updated router-capability in LSDB description.
          * packet counters moved to interface-packet-counters.
          * Added modification information in lsp-log
        ";
      reference "";
    }
    revision 2014-10-24 {
      description
        "
          * Change hello-padding to container
          * Change bfd to container
          * Make BFD a feature
          * Creates mpls-te container and put router-id
          inside
          * Remove GR helper disable and timers
        ";
      reference "draft-ietf-isis-yang-isis-cfg-01";
    }
    revision 2014-10-21 {
      description
        "
         * Interface metric move from af container to interface
          container
         * Hello-padding on interface moved to hello-padding-disable
         with empty type
         * three-way-handshake removed
         * route preference changed to a choice
         * csnp-authentication/psnp-authentication merged
         to authentication container
```

```
          * lsp-gen-interval-exp-delay removed
          * Added overload-max-metric feature
          * overload-max-metric is in a separate container
         ";
       reference "";
     }

     revision 2014-10-07 {
       description
         "
         * Removed spf parameters (should be part of
           vendor specific extensions.
         * Removed hello parameters at global level.
         * Interface configuration uses a string rather
         than a reference. This permits to map to some
         vendor specific configuration.
         ";
       reference "draft-ietf-isis-yang-isis-00";
     }
     revision 2014-09-26 {
       description
         "
         * Add BFD support
         * remove max-elements to max-area-addresses
         ";
       reference "";
     }
     revision 2014-09-11 {
       description
         "
         * Add level parameter to ispf and spf delay
         * Add LSP generation as a feature
         * Make lsp-refresh a feature
         * Change parameter container to list
         ";
       reference "";
     }
     revision 2014-09-05 {
       description
       " Rewrite of the global hierarchy.";
       reference "";
     }
     revision 2014-08-06 {
       description
       "
       * isis-state renamed to isis.
       * Add GR support
       * Add meshgroup support
```

```
        * Add CLNS support
        * Add 64bits tags
        * Add notifications to be aligned with MIB4444
        * Add packet-counters, interface-counters, system-counters
          states
        * Add 3-way handshake support
        * Rename isis-adjacency-updown to adjacency-change
        * Add notification for LSP reception
        * Use feature for reference BW
        * Add lsp-retransmit-interval on interfaces
        * Rename lsp-interval to lsp-pacing-interval
        * Add ispf support as feature
        * Add spf delay support as feature (2step & exp backoff)
        * Add maximum-area-addresses
        * Add default-metric
        ";
        reference "RFC XXXX: YANG Data Model for ISIS Protocol";
      }
      revision 2014-06-25 {
        description "
        * isis-cfg renamed to isis.
        * Add precisions on authentication-keys in description
        ";
        reference "draft-litkowski-isis-yang-isis-01";
      }

      revision 2014-06-20 {
        description "
        * isis-op renamed to isis-state.
        * Multiple instances under ISIS are removed.
        * interface-cfg grouping removed and content
          is directly included in container isis.
        * TLVxx renamed with human-readable name in isis-database.
          TLV reference are putted in description.
        * Reference to core routing module were fixed.
        * Namespace fixed.
        * Add simple-iso-address type.
        * area-id and system-id in ISIS container are merged to
          nsap-address.
        * Add isis-system-id type.
        * Add isis-lsp-id type.
        * Add remaining-lifetime leaf in isis-database.
        * Add TLV2 (is-neighbor) in isis-database.
        * Renamed some container name for consistency
          reason ('isis-' prefixed).
        * Add new identities isis-cfg and isis-state.
        * Add descriptions.
        * Add notification isis-adjacency-updown.
```

```
       * Add RPC clear-isis-adjacency and clear-isis-database.
       ";
       reference "draft-litkowski-isis-yang-isis-00";
     }

     revision 2014-06-11 {
       description "Initial revision.";
       reference "draft-litkowski-netmod-isis-cfg-00";
     }
     identity isis {
       base rt:routing-protocol;
       description "Identity for the ISIS routing protocol.";
     }

     identity isis-adjacency-change {
       description "Identity for the ISIS routing protocol
         adjacency state.";
     }

     identity clear-isis-database {
       description "Identity for the ISIS routing protocol
         database reset action.";
     }

     identity clear-isis-adjacency {
       description "Identity for the ISIS routing protocol
         adjacency reset action.";
     }


     /* Feature definitions */

     feature segment-routing {
       description
        "Support of segment-routing.";
     }
     feature node-tag {
       description
         "Support of node tag.";
     }
     feature igp-ldp-sync {
       description
         "Support of RFC5443.";
     }
     feature fast-reroute {
       description
         "Support of IPFRR.";
     }
```

```
feature  lfa {
  description
    "Support of Loop Free Alternates.";
}
feature remote-lfa {
  description
    "Support of remote Loop Free Alternates.";
}

feature bfd {
  description
    "Support of BFD.";
}
feature overload-max-metric {
  description
    "Support of overload by setting
     all links to max metric.";
}
feature prefix-tag {
  description
    "Add 32bit tag to prefixes";
}
feature prefix-tag64 {
  description
    "Add 64bit tag to prefixes";
}
feature reference-bandwidth {
  description
    "Use a reference bandwidth to compute metric.";
}
feature ipv4-router-id {
  description
   "Support of IPv4 router ID configuration under ISIS.";
}

feature ipv6-router-id {
  description
   "Support of IPv6 router ID configuration under ISIS.";
}

feature multi-topology {
  description
   "Multitopology routing support.";
}
feature nlpid-control {
  description
   "This feature controls the advertisement
    of support NLPID within ISIS configuration.";
```

```
     }
     feature graceful-restart {
       description
         "Graceful restart support as per RFC5306.";
     }


     feature lsp-refresh {
       description
         "Configuration of LSP refresh interval.";
     }

     feature maximum-area-addresses {
       description
         "Support of maximum-area-addresses config.";
     }


     /* Type definitions */


     typedef instance-state-ref {
       type leafref {
         path "/rt:routing-state/rt:routing-instance/"
         +"rt:routing-protocols/rt:routing-protocol/rt:name";
       }
       description
          "This type is used for leaves that reference state data of
           an ISIS protocol instance.";
     }

     typedef admin-state {
       type enumeration {
         enum "up" {
           description
            "Up state";
         }
         enum "down" {
           description
            "Down state";
         }
       }
       description
        "Administrative state of a component.";
     }
     typedef oper-state {
       type enumeration {
         enum "up" {
```

```
          description
           "Up state";
        }
        enum "down" {
          description
           "Down state";
        }
      }
      description
       "Operational state of a component.";
    }
    typedef circuit-id {
      type uint8;
      description
       "This type defines the circuit ID
       associated with an interface.";
    }

    typedef extended-circuit-id {
      type uint32;
      description
       "This type defines the extended circuit ID
       associated with an interface.";
    }

    typedef interface-type {
      type enumeration {
        enum broadcast {
          description "Broadcast interface type.
          Would result in DIS election.";
        }
        enum point-to-point {
          description
           "Point to point interface type.";
        }
      }
      description
       "This type defines the type of adjacency
       to be established on the interface.
       This is affecting the type of hello
       message that would be used.";

    }
    typedef authentication-type {
      type enumeration {
          enum  none {
            description "No authentication used.";
          }
```

```
            enum  plaintext {
              description "Plain text password used.";
            }
            enum  message-digest {
              description "MD5 digest used.";
            }
        }
        description
        "This type defines available authentication types.";
      }

      typedef level {
        type enumeration {
          enum "level-1" {
            description
             "This enum describes L1 only capability.";
          }
          enum "level-2" {
            description
             "This enum describes L2 only capability.";
          }
          enum "level-all" {
            description
             "This enum describes both levels capability.";
          }
        }
        default "level-all";
        description
         "This type defines ISIS level of an object.";

      }

      typedef level-number {
        type uint8 {
          range "1 .. 2";
        }
        description
         "This type defines a current ISIS level.";
      }

      typedef lsp-id {
        type string {
          pattern
           '[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.[0-9A-Fa-f]'
                 +'{4}\.[0-9][0-9]-[0-9][0-9]';
        }
        description
         "This type defines ISIS LSP ID using pattern,
```

```
          system id looks like : 0143.0438.AeF0.02-01";
      }


      typedef area-address {
        type string {
          pattern '[0-9A-Fa-f]{2}\.([0-9A-Fa-f]{4}\.){0,3}';
        }
        description
         "This type defines the area address.";
      }

      typedef snpa {
        type string {
          length "0 .. 20";
        }
        description
         "This type defines Subnetwork Point of Attachement format.";

      }

      typedef system-id {
        type string {
          pattern
           '[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.00';
        }
        description
         "This type defines ISIS system id using pattern,
          system id looks like : 0143.0438.AeF0.00";
      }

      typedef wide-metric {
        type uint32 {
          range "0 .. 16777215";
        }
        description
         "This type defines wide style format
          of ISIS metric.";
      }

      typedef std-metric {
        type uint8 {
          range "0 .. 63";
        }
        description
         "This type defines old style format
          of ISIS metric.";
      }
```

```
typedef mesh-group-state {
  type enumeration {
    enum "meshInactive" {
      description
       "Interface is not part of a mesh group.";
    }
    enum "meshSet" {
      description
       "Interface is part of a mesh group.";
    }
    enum "meshBlocked" {
      description
       "LSPs must not be flooded over that interface.";
    }
  }
  description
   "This type describes meshgroup state of an interface";
}



/* Grouping definitions */

grouping notification-instance-hdr {
  description
   "This group describes common instance specific
    data for notifications.";
  leaf instance-name {
    type string;
    description
     "Describes the name of the ISIS instance.";
  }
  leaf instance-level {
    type level;
    description
     "Describes the ISIS level of the instance.";
  }
}

grouping notification-interface-hdr {
  description
   "This group describes common interface specific
    data for notifications.";
  leaf interface-name {
    type string;
    description
     "Describes the name of the ISIS interface.";
  }
```

```
        leaf interface-level {
          type level;
          description
           "Describes the ISIS level of the interface.";
        }
        leaf extended-circuit-id {
          type extended-circuit-id;
          description
           "Describes the extended circuit-id of the interface.";
        }
      }

      grouping route-content {
        description
         "This group add isis-specific route properties.";
        leaf metric {
          type uint32;
          description
           "This leaf describes ISIS metric of a route.";
        }
        leaf-list tag {
          type uint64;
          description
           "This leaf describes list of tags associated
            with the route. The leaf describes both
            32bits and 64bits tags.";
        }
        leaf route-type {
          type enumeration {
            enum l2-up-internal {
              description "Level 2 internal route
               and not leaked to a lower level";
            }
            enum l1-up-internal {
              description "Level 1 internal route
               and not leaked to a lower level";
            }
            enum l2-up-external {
              description "Level 2 external route
               and not leaked to a lower level";
            }
            enum l1-up-external {
              description "Level 1 external route
               and not leaked to a lower level";
            }
            enum l2-down-internal {
              description "Level 2 internal route
               and leaked to a lower level";
```

```
          }
          enum l1-down-internal {
            description "Level 1 internal route
             and leaked to a lower level";
          }
          enum l2-down-external {
            description "Level 2 external route
             and leaked to a lower level";
          }
          enum l1-down-external {
            description "Level 1 external route
             and leaked to a lower level";
          }
        }
        description
         "This leaf describes the type of ISIS route.";
      }
      leaf segment-id {
        if-feature segment-routing;
        type uint32;
        description
         "Segment ID or MPLS label value
         used to reach the prefix.";
      }
    }

    grouping fast-reroute-global-cfg {
      description
       "This group defines global
       configuration of IPFRR.";
      container fast-reroute {
        if-feature fast-reroute;
        description
          "This container may be
          augmented with global parameters
          for IPFRR.";
          container lfa {
            if-feature lfa;
            description
              "This container may be
              augmented with global parameters
              for LFA.
              Creating the container has no effect on
              LFA activation.";
          }
      }
    }
```

```
     grouping fast-reroute-if-cfg {
       description
        "This group defines interface
        configuration of IPFRR.";
       container fast-reroute {
         if-feature fast-reroute;

         list lfa {
           if-feature lfa;

           key level;

           leaf candidate-disabled {
             type boolean;
             description
              "Prevent the interface to be used as backup.";
           }
           leaf enabled {
             type boolean;
             description
              "Activates LFA.
              This model assumes activation
              of per-prefix LFA.";
           }

           container remote-lfa {
             if-feature remote-lfa;
             leaf enabled {
               type boolean;
               description
               "Activates rLFA.";
             }
             description
              "remote LFA configuration.";
           }
           leaf level {
             type level;
             description
               "Level applicability.";
           }
           description
            "LFA configuration.";
         }
         description
          "Fast-reroute configuration.";
       }
     }
```

```
     grouping segment-routing-global-cfg {
       description
        "This grouping defines SR global config";
       container segment-routing {
         if-feature segment-routing;
         leaf enabled {
           type boolean;
           default false;
           description
            "Enables segment-routing
            protocol extensions.";
         }
         container bindings {
           leaf advertise {
             type boolean;
             default true;
             description
             "Authorize the advertise
             of local mappings in binding TLV.";
           }
           leaf receive {
             type boolean;
             default true;
             description
             "Authorize the reception and usage
             of binding TLV.";
           }
           description
            "Control of binding advertisement
            and reception.";
         }

         description
         "segment routing global config.";
       }
     }

     grouping segment-routing-if-cfg {
       description
        "This grouping defines SR interface config";
       container segment-routing {
         if-feature segment-routing;
         container authorize-bundle {
           leaf enabled {
             type boolean;
             description
              "Enable/disable S flag.";
           }
```

```
          leaf weight {
            type uint8;
            description
             "The value represents the
             weight of the Adj-SID
            for the purpose of load balancing.";
          }
          description
           "Control advertisement of S flag.
           Enable to advertise a common Adj-SID
           for parallel links.";
        }
        leaf advertise-protection {
          type enumeration {
            enum "single" {
              description
               "A single Adj-SID is associated
               with the adjacency and reflects
               the protection configuration.";
            }
            enum "dual" {
              description
               "Two Adj-SIDs will be associated
               with the adjacency if interface
               is protected. In this case
               one will be enforced with
               backup flag set, the other
               will be enforced to backup flag unset.
               In case, protection is not configured,
               a single Adj-SID will be advertised
               with backup flag unset.";
            }
          }
          description
           "If set, the Adj-SID refers to an
          adjacency being protected.";
        }
        description
         "Configuration of segment-routing
          adjacency SID properties.";
      }
    }

    grouping prefix-segment-id {
      description
       "This group defines segment routing extensions
       for prefixes.";
```

```
         leaf flags {
           type bits {
             bit readvertisment {
               position 7;
               description
                "If set, then the prefix to
               which this Prefix-SID is attached,
               has been propagated by the
               router either from another level
               or from redistribution.";
             }
             bit node {
               position 6;
               description
                "If set, then the Prefix-SID refers to
                 the router identified by the prefix.";
             }
             bit php {
               position 5;
               description
               "If set, then the penultimate hop MUST NOT
               pop the Prefix-SID before delivering the packet
               to the node
               that advertised the Prefix-SID.";
             }
             bit explicit-null {
               position 4;
               description
               "If set, any upstream neighbor of
               the Prefix-SID originator MUST replace
               the Prefix-SID with a
               Prefix-SID having an
               Explicit-NULL value (0 for IPv4 and 2 for
               IPv6) before forwarding the packet.";
             }
             bit  value {
               position 3;
               description
               "If set, then the Prefix-SID carries a
               value (instead of an index).
               By default the flag is UNSET.";


             }
             bit  local {
               position 2;
               description
               "If set, then the value/index carried by
```

```
               the Prefix-SID has local significance.
               By default the flag is UNSET.";
             }
           }
           description
            "Describes flags associated with the
            segment ID.";
         }

         leaf algorithm {
           type uint8;
           description
            "Algorithm to be used for path computation.";
         }
         leaf value {
           type uint32;
           description
            "Value of the prefix-SID.";
         }

     }

     grouping adjacency-segment-id {
       description
        "This group defines segment routing extensions
        for adjacencies.";

         leaf flags {
           type bits {
             bit address-family {
               position 7;
               description
                "If unset, then the Adj-SID refers
               to an adjacency with outgoing IPv4 encapsulation.
               If set then the Adj-SID refers to an adjacency
               with outgoing IPv6 encapsulation.";
             }
             bit backup {
               position 6;
               description
                "If set, the Adj-SID refers to an
               adjacency being protected
               (e.g.: using IPFRR or MPLS-FRR)";
             }
             bit  value {
               position 5;
               description
               "If set, then the SID carries a
```

```
               value (instead of an index).
               By default the flag is SET.";


             }
             bit  local {
               position 4;
               description
               "If set, then the value/index carried by
               the SID has local significance.
               By default the flag is SET.";
             }
             bit  set {
               position 3;
               description
               "When set, the S-Flag indicates that the
               Adj-SID refers to a set of adjacencies";
             }
           }

           description
            "Describes flags associated with the
            segment ID.";
         }
         leaf weight {
           type uint8;
           description
            "The value represents the weight of the Adj-SID
             for the purpose of load balancing.";
         }
         leaf neighbor-id {
           type system-id;
           description
            "Describes the system ID of the neighbor
            associated with the SID value. This is only
            used on LAN adjacencies.";
         }
         leaf value {
           type uint32;
           description
            "Value of the Adj-SID.";
         }

     }

     grouping prefix-ipv4-std {
       description
        "This group defines attributes of an
```

```
         IPv4 standard prefix.";
       leaf up-down {
         type boolean;
         description
          "This leaf expresses the value of up/down bit.";
       }
       leaf i-e {
         type boolean;
         description
          "This leaf expresses the value of I/E bit.";
       }
       leaf ip-prefix {
         type inet:ipv4-address;
         description
          "This leaf describes the IPv4 prefix";
       }
       leaf prefix-len {
         type uint8;
         description
          "This leaf describes the IPv4 prefix len in bits";
       }
       leaf default-metric {
         type std-metric;
         description
          "This leaf describes the ISIS default metric value";
       }
       container delay-metric {
         leaf metric {
           type std-metric;
           description
            "This leaf describes the ISIS delay metric value";
         }
         leaf supported {
           type boolean;
           default "false";
           description
            "This leaf describes if the metric is supported.";
         }

         description
          "This container defines the ISIS delay metric.";
       }
       container expense-metric {
         leaf metric {
           type std-metric;
           description
            "This leaf describes the ISIS expense metric value";
         }
```

```
          leaf supported {
            type boolean;
            default "false";
            description
             "This leaf describes if the metric is supported.";
          }
          description
           "This container defines the ISIS expense metric.";
        }
        container error-metric {
          leaf metric {
            type std-metric;
            description
             "This leaf describes the ISIS error metric value";
          }
          leaf supported {
            type boolean;
            default "false";
            description
             "This leaf describes if the metric is supported.";
          }

          description
           "This container defines the ISIS error metric.";
        }
      }

      grouping prefix-ipv4-extended {
        description
         "This group defines attributes of an
          IPv4 extended prefix.";
        leaf up-down {
          type boolean;
          description
           "This leaf expresses the value of up/down bit.";
        }
        leaf ip-prefix {
          type inet:ipv4-address;
          description
           "This leaf describes the IPv4 prefix";
        }
        leaf prefix-len {
          type uint8;
          description
           "This leaf describes the IPv4 prefix len in bits";
        }

        leaf metric {
```

```
          type wide-metric;
          description
           "This leaf describes the ISIS metric value";
        }
        leaf-list tag {
          type uint32;
          description
           "This leaf describes a list of tags associated with
           the prefix.";
        }
        leaf-list tag64 {
          type uint64;
          description
           "This leaf describes a list of 64-bit tags associated with
           the prefix.";
        }
        list prefix-segment-id {
          key value;
          uses prefix-segment-id;
          description
           "List of prefix-segment-id.";
        }
      }

      grouping prefix-ipv6-extended {
        description
         "This group defines attributes of an
          IPv6 prefix.";
        leaf up-down {
          type boolean;
          description
           "This leaf expresses the value of up/down bit.";
        }
        leaf ip-prefix {
          type inet:ipv6-address;
          description
           "This leaf describes the IPv6 prefix";
        }
        leaf prefix-len {
          type uint8;
          description
           "This leaf describes the IPv4 prefix len in bits";
        }

        leaf metric {
          type wide-metric;
          description
           "This leaf describes the ISIS metric value";
```

```
      }
      leaf-list tag {
        type uint32;
        description
         "This leaf describes a list of tags associated with
         the prefix.";
      }
      leaf-list tag64 {
        type uint64;
        description
         "This leaf describes a list of 64-bit tags associated with
         the prefix.";
      }
      list prefix-segment-id {
        key value;
        uses prefix-segment-id;
        description
         "List of prefix-segment-id.";
      }
    }

    grouping neighbor-extended {
      description
       "This group defines attributes of an
        ISIS extended neighbor.";
      leaf neighbor-id {
        type system-id;
        description
         "This leaf describes the system-id of the neighbor.";
      }
      leaf metric {
        type wide-metric;
        description
         "This leaf describes the ISIS metric value";
      }
      list adjacency-segment-id {
        key value;
        uses adjacency-segment-id;
        description
         "List of adjacency-segment-id.";
      }
    }

    grouping neighbor {
      description
       "This group defines attributes of an
        ISIS standard neighbor.";
      leaf neighbor-id {
```

```
          type system-id;
          description
           "This leaf describes the system-id of the neighbor.";
        }
        leaf i-e {
          type boolean;
          description
           "This leaf expresses the value of I/E bit.";
        }
        leaf default-metric {
          type std-metric;
          description
           "This leaf describes the ISIS default metric value";
        }
        container delay-metric {
          leaf metric {
            type std-metric;
            description
             "This leaf describes the ISIS delay metric value";
          }
          leaf supported {
            type boolean;
            default "false";
            description
             "This leaf describes if the metric is supported.";
          }
          description
           "This container defines the ISIS delay metric.";
        }
        container expense-metric {
          leaf metric {
            type std-metric;
            description
             "This leaf describes the ISIS delay expense value";
          }
          leaf supported {
            type boolean;
            default "false";
            description
             "This leaf describes if the metric is supported.";
          }
          description
           "This container defines the ISIS expense metric.";
        }
        container error-metric {
          leaf metric {
            type std-metric;
            description
```

```
            "This leaf describes the ISIS error metric value";
          }
          leaf supported {
            type boolean;
            default "false";
            description
             "This leaf describes if the metric is supported.";
          }
          description
           "This container defines the ISIS error metric.";
        }
      }

      grouping database {
        description
         "This group defines attributes of an
          ISIS database (Link State DB).";
        leaf lsp-id {
          type lsp-id;
          description
           "This leaf describes the LSP ID of the LSP.";
        }
        leaf checksum {
          type uint16;
          description
           "This leaf describes the checksum of the LSP.";
        }
        leaf remaining-lifetime {
          type uint16;
          units "seconds";
          description
           "This leaf describes the remaining lifetime
            in seconds before the LSP expiration.";
        }
        leaf sequence {
          type uint32;
          description
           "This leaf describes the sequence number of the LSP.";
        }
        leaf attributes {
          type bits {
            bit PARTITIONNED {
              description
               "If set, the originator supports partition
                repair.";
            }
            bit ATTACHED-ERROR {
              description
```

```
              "If set, the originator is attached to
               another area using the referred metric.";
            }
            bit ATTACHED-EXPENSE {
              description
               "If set, the originator is attached to
                another area using the referred metric.";
            }
            bit ATTACHED-DELAY {
              description
               "If set, the originator is attached to
                another area using the referred metric.";
            }
            bit ATTACHED-DEFAULT {
              description
               "If set, the originator is attached to
                another area using the referred metric.";
            }
            bit OVERLOAD {
              description
               "If set, the originator is overloaded,
                and must be avoided in path calculation.";
            }
          }
          description
           "This leaf describes attributes of the LSP.";
        }


        container is-neighbor {
          list neighbor {
            key "neighbor-id";
            uses neighbor;
            description
             "List of neighbors.";
          }
          description
           "This leaf describes list of ISIS neighbors.
           ISIS reference is TLV 2.";
        }

        container authentication {
          leaf authentication-type {
            type authentication-type;
            description
             "This leaf describes the authentication type
              to be used.";
          }
```

```
          leaf authentication-key {
            type string;
            description
             "This leaf describes the authentication key
              to be used. For security reason, the
              authentication key MUST NOT be presented
              in plaintext format. Authors recommends
              to use MD5 hash to present the authentication-key.";
          }
          description "This container describes authentication
          information of the node. ISIS reference is TLV 10.";
        }

        container extended-is-neighbor {
          list neighbor {
            key "neighbor-id";
            uses neighbor-extended;
            description
             "List of neighbors.";
          }
          description
           "This container describes list of ISIS extended
           neighbors.
           ISIS reference is TLV 22.";
        }

        container ipv4-internal-reachability {
          list prefixes {
            key "ip-prefix";
            uses prefix-ipv4-std;
            description
             "List of prefixes.";
          }
          description
           "This container describes list of IPv4 internal
           reachability information.
           ISIS reference is TLV 128.";
        }

        leaf-list protocol-supported {
          type uint8;
          description
           "This leaf describes the list of
           supported protocols.
           ISIS reference is TLV 129.";
        }

        container ipv4-external-reachability {
```

```
          list prefixes {
            key "ip-prefix";
            uses prefix-ipv4-std;
            description
             "List of prefixes.";
          }
          description
           "This container describes list of IPv4 external
           reachability information.
           ISIS reference is TLV 130.";
        }

        leaf-list ipv4-addresses {
          type inet:ipv4-address;
          description
            "This leaf describes the IPv4 addresses of the node.
            ISIS reference is TLV 132.";
        }

        leaf ipv4-te-routerid {

          type inet:ipv4-address;
          description
            "This leaf describes the IPv4 Traffic Engineering
            router ID of the node.
            ISIS reference is TLV 134.";
        }

        container extended-ipv4-reachability {

          list prefixes {
            key "ip-prefix";
            uses prefix-ipv4-extended;
            description
             "List of prefixes.";
          }
          description
           "This container describes list of IPv4 extended
           reachability information.
           ISIS reference is TLV 135.";
        }

        leaf dynamic-hostname {
          type string;

          description
           "This leaf describes the name of the node.
           ISIS reference is TLV 137.";
```

```
       }

       leaf ipv6-te-routerid {
         type inet:ipv6-address;
         description
          "This leaf describes the IPv6 Traffic Engineering
           router ID of the node.
           ISIS reference is TLV 140.";
       }


       container mt-is-neighbor {
         list neighbor {
           key "neighbor-id";
           leaf MT-ID {
             type uint16 {
               range "0 .. 4095";
             }
             description
              "This leaf defines the identifier
               of a topology.";
           }
           uses neighbor-extended;
           description
            "List of neighbors.";
         }
         description
          "This container describes list of ISIS multi-topology
           neighbors.
           ISIS reference is TLV 223.";
       }

       container mt-entries {
         list topology {
           key "MT-ID";

           leaf MT-ID {
             type uint16 {
               range "0 .. 4095";
             }
             description
              "This leaf defines the identifier
               of a topology.";
           }

           leaf attributes {
             type bits {
               bit OVERLOAD {
```

```
                 description
                  "If set, the originator is overloaded,
                   and must be avoided in path
                   calculation.";
               }
               bit ATTACHED {
                 description
                  "If set, the originator is attached to
                   another area using the referred metric.";
               }
             }
             description
              "This leaf describes attributes of the LSP
               for the associated topology.";
           }
           description
            "List of topologies supported.";
         }
         description
          "This container describes the topology supported.
          ISIS reference is TLV 229.";
       }

       leaf-list ipv6-addresses {
         type inet:ipv6-address;
         description
          "This leaf describes the IPv6 interface
          addresses of the node.
          ISIS reference is TLV 232.";
       }




       container mt-extended-ipv4-reachability {
         list prefixes {
           key "ip-prefix";
           leaf MT-ID {
             type uint16 {
               range "0 .. 4095";
             }
             description
              "This leaf defines the identifier
               of a topology.";
           }
           uses prefix-ipv4-extended;
           description
            "List of prefixes.";
```

```
          }
          description
           "This container describes list of IPv4
           reachability information in multi-topology
           environment.
           ISIS reference is TLV 235.";
        }

        container mt-ipv6-reachability {
          list prefixes {
            key "ip-prefix";
            leaf MT-ID {
              type uint16 {
                range "0 .. 4095";
              }
              description
               "This leaf defines the identifier
                of a topology.";
            }
            uses prefix-ipv6-extended;
            description
             "List of prefixes.";
          }
          description
           "This container describes list of IPv6
           reachability information in multi-topology
           environment.
           ISIS reference is TLV 237.";
        }

        container ipv6-reachability {
          list prefixes {
            key "ip-prefix";
            uses prefix-ipv6-extended;
            description
             "List of prefixes.";
          }
          description
           "This container describes list of IPv6
           reachability information.
           ISIS reference is TLV 236.";
        }

        list segment-routing-bindings {
          if-feature segment-routing;
          key "fec range";

          leaf fec {
```

```
            type string;
            description
            "IP (v4 or v6) range to be bound to SIDs.";
          }

          leaf range {
            type uint16;
            description
             "Describes number of elements to assign
             a binding to.";
          }

          leaf flags {
            type bits {
              bit address-family {
                position 7;
                description
                "If unset, then the Prefix FEC
                carries an IPv4 Prefix.
                If set then the Prefix FEC carries an
                IPv6 Prefix.";
              }
              bit mirror {
                position 6;
                description
                "Set if the advertised SID/path
                corresponds to a mirrored context.
                ";
              }
              bit flooding {
                position 5;
                description
                 "If the S bit is set(1),
                 the IS-IS Router CAPABILITY TLV
                MUST be flooded across the entire routing domain.
                If the S bit is
                not set(0), the TLV MUST NOT be leaked between levels.
                This bit MUST NOT be altered during the TLV leaking.";
              }
              bit down {
                position 4;
                description
                "When the IS-IS Router CAPABILITY TLV is
                leaked from level-2 to level-1, the D bit
                MUST be set.  Otherwise, this bit MUST
                be clear.  IS-IS Router capability TLVs
                with the D bit set MUST NOT
                be leaked from level-1 to level-2.
```

```
              This is to prevent TLV looping.
              ";
            }
            bit attached {
              position 3;
              description
              "The originator of the SID/Label Binding
              TLV MAY set the A bit in order to signal
              that the prefixes and
              SIDs advertised in the SID/Label Binding
              TLV are directly
              connected to their originators.
              ";
            }

          }
          description
             "Flags of the binding.";
        }

        leaf weight {
          type uint8;
          description
           "Weight of the path for loadbalancing purpose.";
        }

        list binding {
          container prefix-sid {
            uses prefix-segment-id;
            description
             "Binding prefix SID to the range.";
          }
          leaf ero-metric {
            type uint32;
            description
             "Cost of ERO path.";
          }
          container ero {
            leaf address-family {
              type identityref {
                base rt:address-family;
              }
              description
               "Address-family.";
            }

            leaf loose {
              type boolean;
```

```
                description
                 "Set to true,
                 if hop is a loose hop.";
              }
              leaf address {
                type string;
                description
                "IP address of a node on the
                path.";
              }

              description
               "Binding ERO path to the range.";
            }
            container backup-ero {
              leaf address-family {
                type identityref {
                  base rt:address-family;
                }
                description
                 "Address-family.";
              }

              leaf loose {
                type boolean;
                description
                 "Set to true,
                 if hop is a loose hop.";
              }
              leaf address {
                type string;
                description
                "IP address of a node on the
                path.";
              }

              description
               "Binding backup ERO path to the range.";
            }
            container unnumbered-interface-id-ero {
              leaf router-id {
                type string;
                description
                 "Router ID of the node owning the interface.";
              }
              leaf interface-id {
                type uint32;
                description
```

```
              "Interface ID on which the path is built.";
            }
            description
             "Binding a path over unnumbered interface.";
          }
          container backup-unnumbered-interface-id-ero {
            leaf router-id {
              type string;
              description
               "Router ID of the node owning the interface.";
            }
            leaf interface-id {
              type uint32;
              description
              "Interface ID on which the path is built.";
            }
            description
             "Binding a backup path over unnumbered interface.";
          }
          description
           "Bindings associated with the range.";
        }

        description
         "This container describes list of SID/Label
         bindings.
         ISIS reference is TLV 149.";
      }

      list router-capabilities {
        leaf flags {
          type bits {
            bit flooding {
              position 0;
              description
               "If the S bit is set(1),
               the IS-IS Router CAPABILITY TLV
              MUST be flooded across the entire routing domain.
              If the S bit is
              not set(0), the TLV MUST NOT be leaked between levels.
              This bit MUST NOT be altered during the TLV leaking.";
            }
            bit down {
              position 1;
              description
              "When the IS-IS Router CAPABILITY TLV is
              leaked from level-2 to level-1, the D bit
              MUST be set.  Otherwise, this bit MUST
```

```
                be clear.  IS-IS Router capability TLVs
                with the D bit set MUST NOT
                be leaked from level-1 to level-2.
                This is to prevent TLV looping.
                ";
              }
            }
            description
             "Flags associated with router capability.";
          }
          container node-tag {
            if-feature node-tag;
            list node-tag {
              leaf tag {
                type uint32;
                description
                 "Node tag value.";
              }
              description
               "List of tags.";
            }
            description
             "Container for node tags.";
          }
          list segment-routing {
            leaf flags {
              type bits {
                bit ipv4 {
                  position 7;
                  description
                  "If set, then the router is capable of
                  outgoing IPv4 encapsulation on all interfaces.
                  ";

                }
                bit ipv6 {
                  position 6;
                  description
                  "If set, then the router is capable of
                  outgoing IPv6 encapsulation on all interfaces.
                  ";
                }
              }
              description
               "Flags associated with the capability.";
            }
            leaf range {
              type uint32;
```

```
            description
             "Number of values of the range starting with the
             SID/Label value.";
          }
          leaf starting-value {
            type uint32;
            description
            "Lower bound of the label block.";
          }
          description
           "Segment routing capability.";
        }
        leaf-list segment-routing-algorithm {
          type uint8;
          description
           "List of supported algorithm for segment routing.";
        }
        leaf binary {
          type binary;
          description
           "This leaf describes the capability of the node.
           Format is binary according to the protocol encoding.";
        }
        description
         "This container describes the capabilities of the node.
         This container may be extended with detailed
         information.
         ISIS reference is TLV 242.";
      }
    }


    /* */

    augment "/rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route" {
      when "rt:source-protocol = 'isis:isis'" {
        description "ISIS-specific route attributes.";
      }
      uses route-content;
      description
       "This augments route object in RIB with ISIS-specific
        attributes.";
      }

    augment "/rt:active-route/rt:output/rt:route"
       {
      uses route-content;
      description "ISIS-specific route attributes.";
```

```
       }

     augment "/if:interfaces/if:interface"
        {
       leaf clns-mtu {
         type uint16;
         description
          "Defines CLNS MTU of the interface.";
       }
       description "ISO interface config.";
       }


     augment "/rt:routing/rt:routing-instance/rt:routing-protocols/"
         +"rt:routing-protocol" {
       when "rt:type = 'isis:isis'" {
         description
          "This augment is only valid when routing protocol
           instance type is isis.";
       }
       description
        "This augments a routing protocol instance with ISIS
         specific parameters.";
       container isis {

         list instance {
           must "count(area-address) > 0" {
             error-message "At least one area-address
             must be configured.";
             description
              "Enforce configuration of at least one area.";
           }

           key routing-instance;

           leaf routing-instance {
             type rt:routing-instance-ref;
               description
               "Reference routing instance.
               For protocol centric model, which is
               supported in
               default-instance only, this could reference
               any VRF routing-instance.
               For VRF centric model, must reference the
               enclosing routing-instance.";
           }

           leaf level-type {
```

```
              type level;
              default "level-all";
              description
               "This leaf describes the type of ISIS node.
               A node can be level-1-only, level-2-only
               or level-1-2.
               ";
            }

            leaf system-id {
              type system-id;
              description
               "This leaf defines the system-id of the node.";
            }


            leaf maximum-area-addresses {
              if-feature maximum-area-addresses;
              type uint8;
              default 3;
              description
               "Defines the maximum areas supported.";
            }

            leaf-list area-address {
              type area-address;
              description
               "List of areas supported by the
               protocol instance.";
            }

            container mpls {
              leaf ipv4-router-id {
                if-feature ipv4-router-id;
                type inet:ipv4-address;
                description
                 "Router ID value that would be used in
                 TLV 134.";
              }
              leaf ipv6-router-id {
                if-feature ipv6-router-id;
                type inet:ipv6-address;
                description
                 "Router ID value that would be used in
                 TLV 140.";
              }
              container igp-ldp-sync {
                if-feature igp-ldp-sync;
```

```
                description
                  "This container may be augmented
                  with global parameters for igp-ldp-sync.";
              }
              description
                "This container handles mpls config.";
            }
            leaf reference-bandwidth {
              if-feature reference-bandwidth;
              type uint32;
              units "bps";
              description
               "This leaf defines the bandwidth for calculating
                metric.";
            }

            leaf lsp-mtu {
              type uint16;
              units "bytes";
              default 1492;
              description
               "This leaf describes the maximum size of a
                LSP PDU in bytes.";
            }
            leaf lsp-lifetime {
              type uint16;
              units "seconds";
              description
               "This leaf describes the lifetime of the router
                LSP in seconds.";
            }
            leaf lsp-refresh {
              if-feature lsp-refresh;
              type uint16;
              units "seconds";
              description
               "This leaf describes the refresh interval of the
                router LSP in seconds.";
            }

            container graceful-restart {
              if-feature graceful-restart;
              leaf enabled {
                type boolean;
                description
                 "Control enabling the feature.";
              }
              description
```

```
               "This container activates graceful restart.";
            }

            container node-tag {
              if-feature node-tag;

              list node-tag {
                key tag;
                leaf tag {
                  type uint32;
                  description
                   "Node tag value.";
                }
                description
                 "List of tags.";
              }
              description
                "Container for node tags.";
            }

            uses fast-reroute-global-cfg;

            uses segment-routing-global-cfg;


            list authentication {
              key level;

              leaf key {
                type string;
                description
                 "This leaf describes the
                 authentication key.";
              }
              leaf type {
                type authentication-type;
                description
                 "This leaf describes the authentication
                  type to be used.";
              }
              leaf level {
                type level;
                description
                  "Level applicability.";
              }
              description
               "Container for ISIS authentication.
                It covers both LSPs and SNPs.";
```

```
            }



            list metric-type {
              key level;

              leaf value {
                type enumeration {
                  enum wide-only {
                    description
                     "Advertise new metric style only
                     (RFC5305)";
                  }
                  enum old-only {
                    description
                     "Advertise old metric style only
                     (RFC1195)";
                  }
                  enum both {
                    description "Advertise both metric
                    styles";
                  }
                }
                description
                 "This leaf describes the type of metric
                 to be generated.
                 Wide-only means only new metric style
                 is generated,
                 old-only means that only old style metric
                 is generated,
                 and both means that both are advertised.
                 This leaf is only affecting IPv4 metrics.";
              }
              leaf level {
                type level;
                description
                  "Level applicability.";
              }
              description
               "Metric style container.";
            }
            list preference {
              key level;

              choice granularity {
                case detail {
                  leaf internal {
```

```
                type uint8;
                description
                  "This leaf defines the protocol
                   preference for internal routes.";
              }
              leaf external {
                type uint8;
                description
                  "This leaf defines the protocol
                   preference for external routes.";
              }
            }
            case coarse {
              leaf default {
                type uint8;
                description
                  "This leaf defines the protocol
                   preference for all ISIS routes.";
              }
            }
            description
             "Choice for implementation of route preference.";
          }

          leaf level {
            type level;
            description
              "Level applicability.";
          }
          description
             "This leaf defines the protocol preference.";
        }

        list default-metric {
          key level;

          leaf value {
            type wide-metric;
            description
              "Value of the metric";
          }
          leaf level {
            type level;
            description
              "Level applicability of the metric.";
          }
          description
           "Defines the metric to be used by default.";
```

```
            }
            list af {
              if-feature nlpid-control;
              key af;
              leaf af {
                type identityref {
                  base rt:address-family;
                }
                description
                 "Address-family";
              }

              leaf enabled {
                type boolean;
                description
                  "Describes the activation state of the
                   AF.";
              }
              description
               "This list permits activation
               of new address families.";

            }


            list topologies {
              if-feature multi-topology;

              key "name";
              leaf enabled {
                type boolean;
                description
                  "Describes the activation state of the
                   AF.";
              }
              leaf name {
                type rt:rib-ref;
                description "RIB";
              }

              uses fast-reroute-global-cfg;
              uses segment-routing-global-cfg;

              list default-metric {
                key level;

                leaf value {
                  type wide-metric;
```

```
                description
                  "Value of the metric";
              }
              leaf level {
                type level;
                description
                  "Level applicability of the metric.";
              }
              description
               "Defines the metric to be used by default.";
            }
            container node-tag {
              if-feature node-tag;
              list node-tag {
                key tag;
                leaf tag {
                  type uint32;
                  description
                   "Node tag value.";
                }
                description
                 "List of tags.";
              }
              description
                "Container for node tags.";
            }
            description
              "List of topologies";
          }


          list overload {
            key level;

            leaf status {
              type boolean;
              description
               "This leaf defines the overload status.";
            }

            leaf timeout {
              type uint16;
              units "seconds";
              description
               "This leaf defines the timeout in seconds
                of the overload condition.";
            }
            leaf level {
```

```
                type level;
                description
                  "Level applicability of the metric.";
              }
              description
               "This leaf describes if the router is
                set to overload state.";
            }

            list overload-max-metric {
              if-feature overload-max-metric;
              key level;

              leaf status {
                type boolean;
                description
                 "This leaf defines the overload status.";
              }

              leaf timeout {
                type uint16;
                units "seconds";
                description
                 "This leaf defines the timeout in seconds
                  of the overload condition.";
              }
              leaf level {
                type level;
                description
                  "Level applicability of the metric.";
              }
              description
               "This leaf describes if the router is
                set to overload state.";
            }

            container interfaces {
              list interface {
                key "name";
                leaf name {
                  type if:interface-ref;

                  description
                   "Reference to the interface within
                    the routing-instance.";
                }
                leaf level-type {
                  type level;
```

```
                  default "level-all";
                  description
                   "This leaf defines the associated ISIS
                   level of the interface.";
                }
                leaf lsp-pacing-interval {
                  type uint16;
                  units "milliseconds";
                  description
                   "This leaf defines the interval between
                    LSP transmissions in milli-seconds";
                }
                leaf lsp-retransmit-interval {
                  type uint16;
                  units "seconds";
                  description
                   "This leaf defines the interval between
                   retransmission of LSP";
                }
                leaf passive {
                  type boolean;
                  default "false";
                  description
                   "This leaf defines if interface is in
                    passive mode (ISIS not running,
                    but network is advertised).";
                }
                leaf csnp-interval {
                  type uint16;
                  units "seconds";
                  description
                   "This leaf defines the interval of CSNP
                    messages.";
                }

                container hello-padding {
                  leaf enabled {
                    type boolean;
                    default "true";
                    description
                     "Status of Hello-padding activation.
                      By default, the implementation shall
                      pad HELLOs.";
                  }

                  description
                   "This container handles ISIS hello padding
                    configuration.";
```

```
                }

                leaf mesh-group-enabled {
                  type mesh-group-state;
                  description
                   "Describes the mesh group state of
                    the interface.";
                }


                leaf mesh-group {
                  when "../mesh-group-enabled = meshSet" {
                    description
                     "Only valid when mesh-group-enabled
                      equals meshSet";
                  }
                  type uint8;
                  description
                   "Describes the mesh group ID of
                    the interface.";
                }

                leaf interface-type {
                  type interface-type;
                  description
                   "This leaf defines the type of adjacency
                   to be established on the interface.
                   This is affecting the type of hello
                   message that would be used.";
                }

                leaf enabled {
                  type boolean;
                  default "true";
                  description
                   "This leaf describes the administrative
                   status of the ISIS interface.";

                }

                leaf-list tag {
                  if-feature prefix-tag;

                  type uint32;
                  description
                   "This leaf defines list of tags associated
                    with the interface.";
                }
```

```
                leaf-list tag64 {
                  if-feature prefix-tag64;

                  type uint64;
                  description
                   "This leaf defines list of 64bits tags
                   associated with the interface.";
                }

                list hello-authentication {
                  key level;

                  leaf type {
                    type authentication-type;

                    description
                     "This leaf describes the authentication
                     type to be used in hello messages.";
                  }
                  leaf key {
                    type string;
                    description
                     "This leaf describes the
                     authentication key
                     to be used in hello messages.
                     For security reason, the
                     authentication key MUST
                     NOT be presented
                     in plaintext format upon a
                     get-config reply.
                     Authors recommends
                     to use MD5 hash to present the
                     authentication-key";
                  }
                  leaf level {
                    type level;
                    description
                      "Level applicability.";
                  }
                  description
                   "This leaf describes the authentication type
                   to be used in hello messages.";
                }

                list hello-interval {
                  key level;

                  leaf value {
```

```
                    type uint16;
                    units "seconds";
                    description
                     "This leaf defines the interval of
                     hello messages.";
                  }
                  leaf level {
                    type level;
                    description
                      "Level applicability.";
                  }
                  description
                     "This leaf defines the interval of
                     hello messages.";
                }
                list hello-multiplier {
                  key level;

                  leaf value {
                    type uint16;
                    description
                     "This leaf defines the number of
                     hello failed to be received before
                     declaring the adjacency down.";
                  }
                  leaf level {
                    type level;
                    description
                      "Level applicability.";
                  }
                  description
                     "This leaf defines the number of
                      hello failed to be received before
                      declaring the adjacency down.";
                }


                list priority {
                  must 'interface-type = "broadcast"' {
                    error-message
                     "Priority only applies to broadcast
                     interfaces.";
                    description
                     "Check for broadcast interface.";
                  }
                  key level;
                  leaf value {
                    type uint8 {
```

```
                range "0 .. 127";
              }

              description
               "This leaf describes the priority of
               the interface
               for DIS election.";
            }
            leaf level {
              type level;
              description
                "Level applicability.";
            }
            description
               "This leaf describes the priority of
               the interface
               for DIS election.";
          }
          list metric {
            key level;

            leaf value {
              type wide-metric;
              description
               "Metric value.";
            }
            leaf level {
              type level;
              description
                "Level applicability.";
            }
            description
             "Container for interface metric";
          }

          list af {
            key af;

            leaf af {
              type identityref {
                base rt:address-family;
              }
              description
                "Address-family";
            }
            container segment-routing {
              if-feature segment-routing;
              list prefix-sid {
```

```
                    key value;
                    leaf value-type {
                      type enumeration {
                        enum index {
                          description
                             "The value will be
                             interpreted as an index.";

                        }
                        enum absolute {
                          description
                             "The value will become
                             interpreted as an absolute
                             value.";
                        }
                      }
                      description
                        "This leaf defines how value
                         must be interpreted.";
                    }

                    leaf value {
                      type uint32;
                      description
                      "Value associated with
                      prefix. The value must
                      be interpreted in the
                      context of value-type.";
                    }
                    leaf node-flag {
                      type boolean;
                      default true;
                      description
                      "Set prefix as a node
                       representative prefix.";
                    }
                    leaf last-hop-behavior {
                      type enumeration {
                        enum explicit-null {
                          description
                           "Use explicit-null for the SID.";
                        }
                        enum no-php {
                          description
                           "Do no use PHP for the SID.";
                        }
                        enum php {
                          description
```

```
                "Use PHP for the SID.";
              }
            }
            description
             "Configure last hop behavior.";
          }

          description
          "List of prefix-SID associated with
           the interface.";
        }
        description
         "Segment routing interface configuration.";
      }
      container bfd {
        if-feature bfd;
        leaf enabled {
          type boolean;
          default false;
          description
           "This leaf enables BFD.";
        }

        description
          "The container describes
           BFD config.";
      }

      description
       "List of AFs.";
    }

    list topologies {
      key name;

      leaf name {
        type rt:rib-ref;
        description
         "Name of RIB.";
      }
      list metric {
        key level;

        leaf value {
          type wide-metric;
          description
           "Metric value.";
        }
```

```
                  leaf level {
                    type level;
                    description
                      "Level applicability.";
                  }
                  description
                   "Container for interface metric";
                }
                uses fast-reroute-if-cfg;

                uses segment-routing-if-cfg;

                description
                 "List of topologies.";
              }

              uses fast-reroute-if-cfg;

              uses segment-routing-if-cfg;


              container mpls {
                container igp-ldp-sync {
                  if-feature igp-ldp-sync;
                  leaf enabled {
                    type boolean;
                    description
                     "Enable/disable IGP LDP sync.";
                  }
                  description
                   "IGP-LDP sync configuration.";
                }
                description
                 "Container for MPLS specific configuration
                 for ISIS.";
              }
              description
               "List of ISIS interfaces.";
            }


            description
             "This container defines ISIS interface specific
             configuration objects.";
          }
          description
            "List of ISIS instances.";
        }
```

```
         description
          "This container defines ISIS specific configuration
           objects.";
       }
     }

     augment "/rt:routing-state/rt:routing-instance/"
       +"rt:routing-protocols/rt:routing-protocol" {
       when "rt:type = 'isis:isis'" {
         description
          "This augment is only valid when routing protocol
           instance type is isis.";
       }
       description
        "This augments routing protocol instance states with ISIS
         specific parameters.";
       container isis {
         config false;
         list instance {

           key routing-instance;

           leaf routing-instance {
             type rt:routing-instance-ref;
               description
               "Reference routing instance.
               For protocol centric model, which is
               supported in
               default-instance only, this could reference
               any VRF routing-instance.
               For VRF centric model, must reference the
               enclosing routing-instance.";
           }
           container fast-reroute {
             if-feature fast-reroute;

             container protected-routes {
               list af-stats {
                 key "af prefix alternate";

                 leaf af {
                   type identityref {
                     base rt:address-family;
                   }
                   description
                    "Address-family";
                 }
                 leaf prefix {
```

```
                    type string;
                    description
                     "Protected prefix.";
                  }
                  leaf alternate {
                    type string;
                    description
                     "Alternate nexthop for the prefix.";
                  }
                  leaf alternate-type {
                    type enumeration {
                      enum equalcost {
                        description
                         "ECMP alternate.";
                      }
                      enum lfa {
                        description
                         "LFA alternate.";
                      }
                      enum remote-lfa {
                        description
                         "Remote LFA alternate.";
                      }
                      enum tunnel {
                        description
                         "Tunnel based alternate
                         (like RSVP-TE or GRE).";
                      }
                      enum ti-lfa {
                        description
                         "TI LFA alternate.";
                      }
                      enum mrt {
                        description
                         "MRT alternate.";
                      }
                      enum other {
                        description
                         "Unknown alternate type.";
                      }
                    }
                    description
                     "Type of alternate.";
                  }
                  leaf best {
                    type boolean;
                    description
                     "describes if the alternate is the best one.";
```

```
                  }
                  leaf non-best-reason {
                    type string;
                    description
                     "Information field to describe why the alternate
                     is not best.";
                  }
                  leaf protection-available {
                    type bits {
                      bit nodeprotect {
                        position 0;
                        description
                        "Node protection available.";
                      }
                      bit linkprotect {
                        position 1;
                        description
                        "Link protection available.";
                      }
                      bit srlgprotect {
                        position 2;
                        description
                        "SRLG protection available.";
                      }
                      bit downstreamprotect {
                        position 3;
                        description
                        "Downstream protection available.";
                      }
                      bit other {
                        position 4;
                        description
                        "Other protection available.";
                      }
                    }
                    description
                    "Describes protection provided by the alternate.";
                  }
                  leaf alternate-metric1 {
                    type uint32;
                    description
                     "Metric from PLR to destination
                      through the alternate path.";
                  }
                  leaf alternate-metric2 {
                    type uint32;
                    description
                     "Metric from PLR to the alternate node";
```

```
                }
                leaf alternate-metric3 {
                  type uint32;
                  description
                   "Metric from alternate node to the destination";
                }

                description
                 "Per AF statistics.";
              }
              description
               "List of prefixes that are protected.";
            }

            container nonprotected-routes {
               list af-stats {
                key "af prefix";

                leaf af {
                  type identityref {
                    base rt:address-family;
                  }
                  description
                   "Address-family";
                }
                leaf prefix {
                  type string;
                  description
                   "Protected prefix.";
                }
                description
                 "Per AF statistics.";
              }
              description
               "List of prefixes that are not protected.";
            }

            list interfaces {
              key interface;
              leaf interface {
                type string;
                description
                 "This leaf describes the name
                 of the interface.";
              }

              leaf protection-enabled {
                type boolean;
```

```
                  description
                   "This leaf describes if protection
                   is requested for the interface.";
                }

                leaf candidate-disabled {
                  type boolean;
                  description
                   "This leaf describes if interface
                   is prevented to be used as backup.";
                }
                description
                 "Interface fast reroute state.";
              }
              list protection-statistics {
                key frr-protection-method;

                leaf frr-protection-method {
                  type string;
                  description
                   "Protection method used.";
                }
                list af-stats {
                  key af;

                  leaf af {
                    type identityref {
                      base rt:address-family;
                    }
                    description
                     "Address-family";
                  }
                  leaf total-routes {
                    type uint32;
                    description
                     "Total prefixes.";
                  }
                  leaf unprotected-routes {
                    type uint32;
                    description
                      "Total of prefixes who are
                      not protected.";
                  }
                  leaf protected-routes {
                    type uint32;
                    description
                      "Total of prefixes who are
                      protected.";
```

```
                }
                leaf linkprotected-routes {
                  type uint32;
                  description
                    "Total of prefixes who are
                    link protected.";
                }
                leaf nodeprotected-routes {
                  type uint32;
                  description
                    "Total of prefixes who are
                    node protected.";
                }
                description
                 "Per AF statistics.";
              }

              description
               "Global protection statistics.";
            }
            description
             "IPFRR states.";
          }

          container system-counters {
            list level {
              key level;

              leaf level {
                type level-number;
                description
                 "This leaf describes the ISIS level.";
              }
              leaf corrupted-lsps {
                type uint32;
                description
                 "Number of corrupted in-memory LSPs detected.
                 LSPs received from the wire with a bad
                 checksum are silently dropped and not counted.
                 LSPs received from the wire with parse errors
                 are counted by lsp-errors.";
              }
              leaf authentication-type-fails {
                type uint32;
                description
                 "Number of authentication type mismatches.";
              }
              leaf authentication-fails {
```

```
                  type uint32;
                  description
                   "Number of authentication key failures.";
                }
                leaf database-overload {
                  type uint32;
                  description
                   "Number of times the database has become
                   overloaded.";
                }
                leaf own-lsp-purge {
                  type uint32;
                  description
                   "Number of times a zero-aged copy of the
                   system's own LSP is received from some
                   other node.";
                }
                leaf manual-address-drop-from-area {
                  type uint32;
                  description
                   "Number of times a manual address
                    has been dropped from the area.";
                }
                leaf max-sequence {
                  type uint32;
                  description
                   "Number of times the system has attempted
                    to exceed the maximum sequence number.";
                }
                leaf sequence-number-skipped {
                  type uint32;
                  description
                   "Number of times a sequence number skip has
                   occured.";
                }
                leaf id-len-mismatch {
                  type uint32;
                  description
                   "Number of times a PDU is received with
                    a different value for ID field length
                    from that of the receiving system.";
                }
                leaf partition-changes {
                  type uint32;
                  description
                   "Number of partition changes detected.";
                }
                leaf lsp-errors {
```

```
                type uint32;
                description
                 "Number of LSPs with errors we have
                 received.";
              }
              leaf spf-runs {
                type uint32;
                description
                 "Number of times we ran SPF at this level.";
              }
              description
               "List of supported levels.";
            }
            description
             "The container defines a list of counters
              for the IS.";
          }

          container interfaces {
            list interfaces {
              key interface;

              leaf interface {
                type string;
                description
                 "This leaf describes the name
                 of the interface.";
              }
              container status {
                leaf circuit-id {
                  type circuit-id;
                  description
                   "This leaf describes the circuit-id
                    associated with the interface.";
                }
                leaf admin-state {
                  type admin-state;
                  description
                  "This leaf describes the admin state
                  of the interface.";
                }
                leaf oper-state {
                  type oper-state;
                  description
                  "This leaf describes the operational state
                  of the interface.";
                }
                leaf interface-type {
```

```
                  type interface-type;
                  description
                  "Type of interface to be used.";
                }
                leaf passive {
                  type empty;
                  description
                   "The interface is included in LSP, but
                   does not run ISIS protocol.";
                }
                leaf three-way-handshake {
                  type boolean;
                  description
                   "The interface uses 3-way handshake.";
                }
                leaf mesh-group-enabled {
                  type mesh-group-state;
                  description
                   "Mesh group status.";
                }
                leaf mesh-group {
                  type uint8;
                  description
                   "Mesh group value.";
                }
                list level {
                  key level;

                  leaf level {
                    type level;
                    description
                    "Level associated with the interface.";
                  }
                  leaf priority {
                    type uint8;
                    description
                     "Priority for becoming the LAN DIS.";
                  }
                  leaf hello-multiplier {
                    type uint16;
                    description
                     "Hello multiplier.";
                  }
                  leaf hello-interval {
                    type uint16;
                    description
                     "Hello interval in msec.";
                  }
```

```
                leaf dr-hello-interval {
                  type uint32;
                  description
                   "Hello interval used by DIS in msec.";
                }

                leaf metric {
                  type empty;
                  description
                   "The interface is included in LSP, but
                   does not run ISIS protocol.";
                }
                description
                 "Per level states.";
              }

              description
               "Provides ISIS interface status parameters.";
            }
            container event-counters {
              leaf adjacency-changes {
                type uint32;
                description
                 "The number of times an adjacency state
                  change has occured on this interface.";
              }
              leaf adjacency-number {
                type uint32;
                description
                 "The number of adjacencies on this
                  interface.";
              }
              leaf init-fails {
                type uint32;
                description
                 "The number of times initialization of
                  this interface has failed. This counts
                  events such as PPP NCP failures.
                  Failures to form an adjacency are counted
                  by adjacency-rejects.";
              }
              leaf adjacency-rejects {
                type uint32;
                description
                 "The number of times an adjacency has been
                  rejected on this interface.";
              }
              leaf id-len-mismatch {
```

```
                type uint32;
                description
                 "The number of times an IS-IS PDU with an ID
                  field length different from that for this
                  system has been received on this interface.";
              }
              leaf max-area-addresses-mismatch {
                type uint32;
                description
                 "The number of times an IS-IS PDU with
                  according max area address field
                  differs from that for
                  this system has been received on this
                  interface.";
              }
              leaf authentication-type-fails {
                type uint32;
                description
                 "Number of authentication type mismatches.";
              }
              leaf authentication-fails {
                type uint32;
                description
                 "Number of authentication key failures.";
              }
              leaf lan-dis-changes {
                type uint32;
                description
                 "The number of times the DIS has changed
                  on this interface at this level.
                  If the interface type is point to point,
                  the count is zero.";
              }
              description
               "Provides protocol event counters.";
            }
            container packet-counters {
              list level {
                key level;

                leaf level {
                  type level-number;
                  description
                   "This leaf describes the ISIS level.";
                }

                container iih {
                  leaf in {
```

```
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of IIH PDUs received/sent.";
                }
                container ish {
                  leaf in {
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of ISH PDUs received/sent.";
                }
                container esh {
                  leaf in {
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of ESH PDUs received/sent.";
                }
                container lsp {
                  leaf in {
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
```

```
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of LSP PDUs received/sent.";
                }
                container psnp {
                  leaf in {
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of PSNP PDUs received/sent.";
                }
                container csnp {
                  leaf in {
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of CSNP PDUs received/sent.";
                }
                container unknown {
                  leaf in {
                    type uint32;
                    description
                     "Received PDUs.";
                  }
                  leaf out {
                    type uint32;
                    description
                     "Sent PDUs.";
                  }
                  description
                   "The number of unknown PDUs received/sent.";
                }
```

```
                 description
                  "List of supported levels.";
               }
               description
                "Provides packet counters per level.";
             }
             description
              "List of interfaces.";
           }
           description
            "The container defines operational parameters
             of interfaces.";
         }
         container adjacencies {
           list adjacency {
             leaf interface {
               type string;
               description
                "This leaf describes the name
                of the interface.";
             }
             leaf neighbor-systype {
               type level;
               description
                "Type of neighboring system";

             }
             leaf neighbor-sysid {
               type system-id;
               description
                "The system-id of the neighbor";

             }
             leaf neighbor-extended-circuit-id {
               type extended-circuit-id;
               description
                "Circuit ID of the neighbor";
             }
             leaf neighbor-snpa {
               type snpa;
               description
                "SNPA of the neighbor";
             }
             leaf usage {
               type level;
               description
               "How is the adjacency used ?
               On a p2p link this might be level 1 and 2,
```

```
                but on a LAN, the usage will be level 1
                between peers at L1 or level 2 between
                peers at L2.";
              }
              leaf hold-timer {
                type uint16;
                description
                 "The holding time in seconds for this
                  adjacency. This value is based on
                  received hello PDUs and the elapsed
                  time since receipt.";
              }
              leaf neighbor-priority {
                type uint8 {
                  range "0 .. 127";
                }
                description
                 "Priority of the neighboring IS for becoming
                  the DIS.";
              }
              leaf lastuptime {
                type yang:timestamp;
                description
                 "When the adjacency most recently entered
                  state 'up', measured in hundredths of a
                  second since the last reinitialization of
                  the network management subsystem.
                  The value is 0 if the adjacency has never
                  been in state 'up'.";

              }
              leaf state {
                type enumeration {
                  enum "Up" {
                    description
                     "This state describes that
                      adjacency is established.";
                  }
                  enum "Down" {
                    description
                     "This state describes that
                      adjacency is NOT established.";
                  }
                  enum "Init" {
                    description
                     "This state describes that
                      adjacency is establishing.";
                  }
```

```
                enum "Failed" {
                  description
                   "This state describes that
                    adjacency is failed.";
                }
              }
              description
               "This leaf describes the state of the
               interface.";
            }
            list adjacency-sid {
              if-feature segment-routing;
              key value;
              leaf af {
                type identityref {
                  base rt:address-family;
                }
                description
                 "Address-family associated with the
                 segment ID";
              }
              leaf value {
                type uint32;
                description
                 "Value of the Adj-SID.";
              }
              leaf weight {
                type uint8;
                description
                 "Weight associated with
                 the adjacency SID.";
              }
              leaf protection-requested {
                type boolean;
                description
                 "Describe if the adjacency SID
                 must be protected.";
              }
              description
               "List of adjacency Segment IDs.";
            }
            description
             "List of operational adjacencies.";
          }
          description
           "This container lists the adjacencies of
           the local node.";
        }
```

```
            container spf-log {
              list event {
                key id;

                leaf id {
                  type uint32;
                  description
                   "This leaf defines the event identifier.
                    This is a purely internal value.";
                }
                leaf spf-type {
                  type enumeration {
                    enum full {
                      description
                       "Computation done is a Full SPF.";
                    }
                    enum incremental {
                      description
                       "Computation done is an
                        incremental SPF.";
                    }
                    enum route-only {
                      description
                       "Computation done is a
                        reachability computation
                        only.";
                    }
                  }
                  description
                   "This leaf describes the type of computation
                   used.";
                }
                leaf level {
                  type level-number;
                  description
                   "This leaf describes the level affected by the
                   the computation.";
                }
                leaf spf-delay {
                  type uint32;
                  units "milliseconds";
                  description
                   "This leaf describes the SPF delay that
                   was used for this event.";
                }
                leaf schedule-timestamp {
                  type yang:timestamp;
                  description
```

```
                 "This leaf describes the timestamp
                  when the computation was scheduled.";
               }
               leaf start-timestamp {
                 type yang:timestamp;
                 description
                  "This leaf describes the timestamp
                   when the computation was started.";
               }
               leaf end-timestamp {
                 type yang:timestamp;
                 description
                  "This leaf describes the timestamp
                   when the computation was ended.";
               }
               list trigger-lsp {
                 key "lsp";
                 leaf lsp {
                   type lsp-id;
                   description
                    "This leaf describes the LSPID
                     of the LSP.";
                 }
                 leaf sequence {
                   type uint32;
                   description
                    "This leaf describes the sequence
                     number of the LSP.";
                 }
                 description
                  "This leaf describes list of LSPs
                   that triggered the computation.";
               }
               description
                "List of computation events.";
             }

             description
              "This container lists the SPF computation events.";
           }
           container lsp-log {
             list event {
               key id;

               leaf id {
                 type uint32;
                 description
                  "This leaf defines the event identifier.
```

```
                    This is a purely internal value.";
                }
                leaf level {
                  type level-number;
                  description
                   "This leaf describes the level affected by the
                   the computation.";
                }
                container lsp {
                  leaf lsp {

                    type lsp-id;
                    description
                     "This leaf describes the LSPID
                      of the LSP.";
                  }
                  leaf sequence {
                    type uint32;
                    description
                     "This leaf describes the sequence
                      number of the LSP.";
                  }
                  description
                   "This container describes the received LSP
                   , in case of local LSP update the local
                   LSP ID is referenced.";
                }

                leaf received-timestamp {
                  type yang:timestamp;

                  description
                   "This leaf describes the timestamp
                   when the LSP was received. In case of
                   local LSP update, the timestamp refers
                   to the local LSP update time.";
                }

                leaf change {
                  type bits {
                    bit refresh {
                      position 0;
                      description
                       "Refresh LSP, nothing has changed.";
                    }
                    bit link-down {
                      position 1;
                      description
```

```
                    "One or more links are down.";
                  }
                  bit link-up {
                    position 2;
                    description
                     "One or more links are up.";
                  }
                  bit link-metric-change {
                    position 3;
                    description
                     "One or more links experienced
                     a metric change.";
                  }
                  bit link-other-change {
                    position 4;
                    description
                     "One or more links experienced
                     a change that does not affect state
                     or metric.";
                  }
                  bit prefix-down {
                    position 5;
                    description
                     "One or more links are down.";
                  }
                  bit prefix-up {
                    position 6;
                    description
                     "One or more prefixes are up.";
                  }
                  bit prefix-metric-change {
                    position 7;
                    description
                     "One or more prefixes experienced
                     a metric change.";
                  }
                  bit prefix-other-change {
                    position 8;
                    description
                     "One or more prefixes experienced
                     a change that does not affect state
                     or metric.";
                  }
                  bit other-change {
                    position 9;
                    description
                     "One or more component changed that
                     is not a prefix or link.";
```

```
                }
              }
              description
               "This leaf describes the type of change
               in the LSP.";
            }

            description
             "List of LSP events.";
          }

          description
           "This container lists the LSP reception events.
           Local LSP modification are also contained in the
           list.";
        }
        container database {
          list level-db {
            key level;

            leaf level {
              type level-number;
              description
               "Current level number";
            }
            list lsp {
              key lsp-id;

              uses database;
              description
               "List of LSPs in LSDB.";
            }

            description
             "This container describes the list of LSPs
             in the level x database.";
          }

          description
           "This container describes ISIS Link State
            databases.";
        }
        container hostnames {

          list hostname {
            key system-id;
            leaf system-id {
              type system-id;
```

```
                 description
                  "This leaf describes the system-id
                  associated with the hostname.";
               }
               leaf hostname {

                 type string;
                 description
                  "This leaf describes the hostname
                   associated with the system ID.";
               }
               description
                "List of system-id/hostname associations";
             }

             description
              "This container describes the list
              of binding between system-id and
              hostnames.";
           }

           description
            "List of ISIS instances.";
         }
         description
            "This container defines various ISIS states objects.";
       }
     }

     /* RPC methods */

     rpc clear-adjacency {
       description
        "This RPC request clears a particular
         set of ISIS adjacencies. If the operation
         fails for ISIS internal reason, then
         error-tag and error-app-tag should be set
         to a meaningful value.";
       input {
                       leaf routing-instance-name {
           type rt:routing-instance-state-ref;
           mandatory "true";
           description
            "Name of the routing instance whose ISIS
             information is being queried.

             If the routing instance with name equal to the
             value of this parameter doesn't exist, then this
```

```
             operation SHALL fail with error-tag 'data-missing'
             and error-app-tag 'routing-instance-not-found'.";


         }
         leaf routing-protocol-instance-name {
           type instance-state-ref;
           mandatory "true";
           description
            "Name of the ISIS protocol instance whose ISIS
             information is being queried.

             If the ISIS instance with name equal to the
             value of this parameter doesn't exist, then this
             operation SHALL fail with error-tag 'data-missing'
             and error-app-tag
             'routing-protocol-instance-not-found'.";
         }
         leaf level {
           type level;
           description
            "ISIS level of the adjacency to be cleared.
            If ISIS level is level-1-2, both level 1 and level 2
            adjacencies would be cleared.

            If the value provided is different from the one
            authorized in the enum type, then this
             operation SHALL fail with error-tag 'data-missing'
             and error-app-tag
             'bad-isis-level'.
            ";
         }
         leaf interface {
           type string;
           description
            "Name of the ISIS interface.

             If the ISIS interface with name equal to the
             value of this parameter doesn't exist, then this
             operation SHALL fail with error-tag 'data-missing'
             and error-app-tag
             'isis-interface-not-found'.";
         }
       }
     }

     rpc clear-database {
       description
```

```
         "This RPC request clears a particular
          ISIS database. If the operation
          fails for ISIS internal reason, then
          error-tag and error-app-tag should be set
          to a meaningful value.";
        input {
          leaf routing-instance-name {
            type rt:routing-instance-state-ref;
            mandatory "true";
            description
             "Name of the routing instance whose ISIS
              information is being queried.

              If the routing instance with name equal to the
              value of this parameter doesn't exist, then this
              operation SHALL fail with error-tag 'data-missing'
              and error-app-tag 'routing-instance-not-found'.";


          }
          leaf routing-protocol-instance-name {
            type instance-state-ref;
            mandatory "true";
            description
             "Name of the ISIS protocol instance whose ISIS
              information is being queried.

              If the ISIS instance with name equal to the
              value of this parameter doesn't exist, then this
              operation SHALL fail with error-tag 'data-missing'
              and error-app-tag
              'routing-protocol-instance-not-found'.";
          }
          leaf level {
            type level;
            description
             "ISIS level of the adjacency to be cleared.
             If ISIS level is level-1-2, both level 1 and level 2
             adjacencies would be cleared.

             If the value provided is different from the one
             authorized in the enum type, then this
              operation SHALL fail with error-tag 'data-missing'
              and error-app-tag
              'bad-isis-level'.
             ";
          }
        }
```

```
     }



     /* Notifications */

     notification database-overload {
       uses notification-instance-hdr;

       leaf overload {
         type enumeration {
           enum "off" {
             description
              "The system has left overload condition.";
           }
           enum "on" {
             description
              "The system is in overload condition.";
           }

         }
         description
            "Describes the new overload state of the instance.";
       }
       description
         "This notification is sent when an ISIS instance
          overload condition changes.";
     }

     notification lsp-too-large {
       uses notification-instance-hdr;
       uses notification-interface-hdr;

       leaf pdu-size {
         type uint32;
         description
          "Size of the PDU";
       }
       leaf lsp-id {
         type lsp-id;
         description
          "LSP ID.";
       }
       description
         "This notification is sent when we attempt
          to propagate an LSP that is larger than the
          dataLinkBlockSize for the circuit.
          The notification generation must be throttled
```

```
          with at least a 5 second gap.
          ";
     }

     notification corrupted-lsp-detected {
       uses notification-instance-hdr;
       leaf lsp-id {
         type lsp-id;
         description
          "LSP ID.";
       }
       description
         "This notification is sent when we find
          that an LSP that was stored in memory has
          become corrupted.
          ";
     }

     notification attempt-to-exceed-max-sequence {
       uses notification-instance-hdr;
       leaf lsp-id {
         type lsp-id;
         description
          "LSP ID.";
       }
       description
         "This notification is sent when the system
         wraps the 32-bit sequence counter of an LSP.
          ";
     }

     notification id-len-mismatch {
       uses notification-instance-hdr;
       uses notification-interface-hdr;

       leaf pdu-field-len {
         type uint8;
         description
          "Size of the ID length in the received PDU";
       }
       leaf raw-pdu {
         type binary;
         description
          "Received raw PDU.";
       }
       description
         "This notification is sent when we receive a PDU
         with a different value for the System ID length.
```

```
         The notification generation must be throttled
         with at least a 5 second gap.
          ";
     }

     notification max-area-addresses-mismatch {
       uses notification-instance-hdr;
       uses notification-interface-hdr;

       leaf max-area-addresses {
         type uint8;
         description
          "Received number of supported areas";
       }
       leaf raw-pdu {
         type binary;
         description
          "Received raw PDU.";
       }
       description
         "This notification is sent when we receive a PDU
         with a different value for the Maximum Area Addresses.
         The notification generation must be throttled
         with at least a 5 second gap.
          ";
     }

     notification own-lsp-purge {
       uses notification-instance-hdr;
       uses notification-interface-hdr;
       leaf lsp-id {
         type lsp-id;
         description
          "LSP ID.";
       }
       description
         "This notification is sent when the system
         receives a PDU with its own system ID and zero age.
          ";
     }

     notification sequence-number-skipped {
       uses notification-instance-hdr;
       uses notification-interface-hdr;
       leaf lsp-id {
         type lsp-id;
         description
          "LSP ID.";
```

```
      }
      description
        "This notification is sent when the system
        receives a PDU with its own system ID and
        different contents. The system has to reissue
        the LSP with a higher sequence number.
         ";
    }

    notification authentication-type-failure {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      description
        "This notification is sent when the system
        receives a PDU with the wrong authentication type
        field.
        The notification generation must be throttled with
        at least a 5 second gap.
         ";
    }

    notification authentication-failure {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      description
        "This notification is sent when the system
        receives a PDU with the wrong authentication
        information.
        The notification generation must be throttled with
        at least a 5 second gap.
         ";
    }

    notification version-skew {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf protocol-version {
        type uint8;
```

```
        description
         "Protocol version received in the PDU.";
      }
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      description
        "This notification is sent when the system
        receives a PDU with a different protocol version
        number.
        The notification generation must be throttled with at least
         a 5 second gap.
         ";
    }

    notification area-mismatch {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      description
        "This notification is sent when the system
        receives a Hello PDU from an IS that does
        not share any area address.
        The notification generation must be throttled with at least
         a 5 second gap.
         ";
    }

    notification rejected-adjacency {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      leaf reason {
        type string;
        description
         "The system may provide a reason to reject the
          adjacency. If the reason is not available,
          the system use an empty string.";
```

```
      }
      description
        "This notification is sent when the system
        receives a Hello PDU from an IS but does not
        establish an adjacency for some reason.
        The notification generation must be throttled with at least
         a 5 second gap.
         ";
    }


    notification protocols-supported-mismatch {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      leaf-list protocols {
        type uint8;
        description
         "The list of protocols supported by the
          remote system.";
      }
      description
        "This notification is sent when the system
        receives a non pseudonode LSP that has no matching
        protocol supported.
        The notification generation must be throttled with at least
         a 5 second gap.
         ";
    }

    notification lsp-error-detected {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf lsp-id {
        type lsp-id;
        description
         "LSP ID.";
      }
      leaf raw-pdu {
        type binary;
        description
         "Received raw PDU.";
      }
      leaf error-offset {
```

```
        type uint32;
        description
         "If the problem is a malformed TLV,
         the error-offset points to the start of the TLV.
         If the problem is with the LSP header,
         the error-offset points to the suspicious byte";
      }
      leaf tlv-type {
        type uint8;
        description
         "if the problem is a malformed TLV, the tlv-type is set
         to the type value of the suspicious TLV.
         Otherwise this leaf is not present.";
      }
      description
        "This notification is sent when the system
        receives a  LSP with a parse error.
        The notification generation must be throttled with at least
         a 5 second gap.
         ";
    }

    notification adjacency-change {
      uses notification-instance-hdr;
      uses notification-interface-hdr;
      leaf neighbor {
        type string;
        description
         "Describes the name of the neighbor. If the
         name of the neighbor is not available, the
         field would be empty.";
      }
      leaf neighbor-system-id {
        type system-id;
        description
         "Describes the system-id of the neighbor.";
      }
      leaf level {
        type level;
        description
         "Describes the ISIS level of the adjacency.";
      }
      leaf state {
        type enumeration {
          enum "Up" {
            description
             "This state describes that
              adjacency is established.";
```

```
            }
            enum "Down" {
              description
               "This state describes that
                adjacency is no more established.";
            }
          }
          description
           "This leaf describes the new state of the
            ISIS adjacency.";
        }
        leaf reason {
          type string;
          description
           "If the adjacency is going to DOWN,
           this leaf provides a reason for the adjacency
           going down. The reason is provided as a text.
           If the adjacency is going to UP, no reason is
           provided.";
        }
        description
          "This notification is sent when an ISIS adjacency
           moves to Up state or to Down state.";
      }

      notification lsp-received {
        uses notification-instance-hdr;
        uses notification-interface-hdr;

        leaf lsp-id {
          type lsp-id;
          description
           "LSP ID.";
        }
        leaf sequence {
          type uint32;
          description
           "Sequence number of the received LSP.";
        }
        leaf received-timestamp {
          type yang:timestamp;

          description
           "This leaf describes the timestamp
           when the LSP was received. ";
        }
        leaf neighbor-system-id {
          type system-id;
```

```
        description
         "Describes the system-id of the neighbor
         that sent the LSP.";
      }
      description
        "This notification is sent when a LSP
        is received.
        The notification generation must be throttled with at least
         a 5 second gap. ";
    }

    notification lsp-generation {
      uses notification-instance-hdr;

      leaf lsp-id {
        type lsp-id;
        description
         "LSP ID.";
      }
      leaf sequence {
        type uint32;
        description
         "Sequence number of the received LSP.";
      }
      leaf send-timestamp {
        type yang:timestamp;

        description
         "This leaf describes the timestamp
         when our LSP was regenerated. ";
      }
      description
        "This notification is sent when a LSP
        is regenerated.
        The notification generation must be throttled with at least
         a 5 second gap. ";
    }


  }

  <CODE ENDS>
```

7. Security Considerations

Configuration and state data defined in this document are designed to be accessed via the NETCONF protocol [RFC6241].

As ISIS is an IGP protocol (critical piece of the network), ensuring stability and security of the protocol is mandatory for the network service.

Authors recommends to implement NETCONF access control model ([RFC6536]) to restrict access to all or part of the configuration to specific users. Access control to RPCs is also critical as RPC permits to clear protocol datastructures that would definitively impact the network service. This kind of RPC needs only to be used in specific cases by well-known experienced users.

Authors consider that all the configuration is considered as sensitive/vulnerable as well as RPCs. But security teams can decide to open some part of the configuration to less experienced users depending on the internal organization, for example:

- User FullWrite: would access to the whole data model. This kind of profile may be restricted to few experienced people.

- User PartialWrite: would only access to configuration part within /isis/interfaces/interface. So this kind of profile is restricted to creation/modification/deletion of interfaces. This profile does not have access to RPC.

- User Read: would only access to state part /isis-state.

Unauthorized access to configuration or RPC may cause high damages to the network service.

The /isis-state/database may contain authentication information. As presented in the description of the /isis-state/database/level-1/lsp/authentication/authentication-key, the authentication MUST never be presented in plaintext format for security reason. Authors recommends the usage of MD5 to present the authentication-key.

Some authentication-key may also be present in the /isis configuration. When configuring ISIS using the NETCONF protocol, authors recommends the usage of secure transport of NETCONF using SSH ([RFC6242]).

8. Contributors

Authors would like to thank Kiran Agrahara Sreenivasa, Dean Bogdanovic, Yingzhen Qu, Yi Yang for their major contributions to the draft.

9. Acknowledgements

TBD.

10. IANA Considerations

TBD.

11. Normative References

[I-D.ietf-netmod-routing-cfg]
Lhotka, L., "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-16 (work in progress), October 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, October 2010.

[RFC6241] Enns, R., Bjorklund, M., Schoenwaelder, J., and A. Bierman, "Network Configuration Protocol (NETCONF)", RFC 6241, June 2011.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, June 2011.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, March 2012.

Appendix A. Example: NETCONF <get> Reply

This section gives an example of a reply to the NETCONF <get> request for a device that implements the data model defined in this document. The example is written in XML.

```
<?xml version="1.0" encoding="utf-8"?>
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <routing xmlns="urn:ietf:params:xml:ns:yang:ietf-routing">
    <routing-instance>
      <name>SLI</name>
      <router-id>1.1.1.1</router-id>
      <description/>
      <default-ribs>
        <default-rib>
```

```
          <address-family>ipv4-unicast</address-family>
          <rib-name>default</rib-name>
        </default-rib>
      </default-ribs>
      <interfaces>
        <interface>
          <name>Loopback0</name>
        </interface>
                <interface>
          <name>Eth1</name>
        </interface>
      </interfaces>
      <routing-protocols>
        <routing-protocol>
          <name>ISIS</name>
          <description/>
          <type>isis:isis</type>
          <connected-ribs>
            <connected-rib>
              <rib-name>default</rib-name>
              <import-filter/>
              <export-filter/>
            </connected-rib>
          </connected-ribs>
          <isis xmlns="urn:ietf:params:xml:ns:yang:ietf-isis">
            <instance>
              <routing-instance>SLI</routing-instance>
                          <level-type>level-2</level-type>
              <system-id>87FC.FCDF.4432</system-id>
              <area-address>49.0001</area-address>
              <mpls-te>
                <ipv4-router-id>1.1.1.1</ipv4-router-id>
              </mpls-te>
              <lsp-lifetime>65535</lsp-lifetime>
              <lsp-refresh>65000</lsp-refresh>
              <authentication>
                            <key>ThisIsThePassword</key>
                <type>plain-text</type>
                                <level>level-2</level>
              </authentication>
              <metric-type>
                <value>wide</value>
              </metric-type>
              <default-metric>
                <value>111111</value>
              </default-metric>
              <af>
                <af>ipv4-unicast</af>
```

```
                <enabled>true</enabled>
              </af>
              <interfaces>
                <interface>
                  <name>Loopback0</name>
                  <tag>200</tag>
                  <metric>
                    <value>0</value>
                  </metric>
                                  <passive>true</passive>
                </interface>
                                <interface>
                  <name>Eth1</name>
                                  <level-type>level-2</level-type>
                                  <interface-type>point-to-point</interface-
type>
                  <metric>
                    <value>167890</value>
                  </metric>

                </interface>
              </interfaces>
            </instance>
          </isis>
        </routing-protocol>
      </routing-protocols>
    </routing-instance>
    <ribs>
      <rib>
        <name>default</name>
        <address-family>ipv4-unicast</address-family>
        <description/>
        <recipient-ribs>
          <recipient-rib>
            <rib-name/>
            <filter/>
          </recipient-rib>
        </recipient-ribs>
      </rib>
    </ribs>
    <route-filters>
      <route-filter>
        <name/>
        <description/>
        <type/>
      </route-filter>
    </route-filters>
  </routing>
```

```
  <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
```

```
    <interface>
      <name>Loopback0</name>
      <description/>
      <type/>
      <link-up-down-trap-enable/>
      <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
        <mtu/>
        <address>
          <ip>1.1.1.1</ip>
          <prefix-length>32</prefix-length>
        </address>
      </ipv4>

    </interface>
        <interface>
      <name>Eth1</name>
      <description/>
      <type/>
      <link-up-down-trap-enable/>
      <ipv4 xmlns="urn:ietf:params:xml:ns:yang:ietf-ip">
        <mtu/>
        <address>
          <ip>10.0.0.1</ip>
          <prefix-length>30</prefix-length>
        </address>
      </ipv4>

    </interface>
  </interfaces>
</data>
```


Authors' Addresses

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Derek Yeung
Cisco Systems

Email: myeung@cisco.com

Acee Lindem
Cisco Systems

Email: acee@cisco.com

Jeffrey Zhang
Juniper Networks

Email: zzhang@juniper.net

Ladislav Lhotka

Email: lhotka@nic.cz