

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2015

L. Ginsberg
Cisco Systems
S. Litkowski
Orange Business Service
S. Previdi
Cisco Systems
July 02, 2014

IS-IS Route Preference for Extended IP and IPv6 Reachability
draft-ginsberg-isis-route-preference-00.txt

Abstract

Existing specifications as regards route preference are not explicit when applied to IP/IPv6 Extended Reachability TLVs. There are also inconsistencies in the definition of how the up/down bit applies to route preference when the prefix advertisement appears in Level 2 LSPs. This document addresses these issues.

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 RFC 2119 [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 January 3, 2015.

Copyright Notice

Copyright (c) 2014 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.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	2
2. Use of the up/down Bit in Level 2 LSPs	3
3. Types of Routes in IS-IS Supported by Extended Reachability TLVs	4
3.1. Types of Routes Supported by TLVs 135 and 235	4
3.2. Types of Routes Supported by TLVs 236 and 237	5
3.3. Order of Preference for all types of routes supported by TLVs 135 and 235	6
3.4. Order of Preference for all types of routes supported by TLVs 236 and 237	7
4. IANA Considerations	7
5. Security Considerations	7
6. Acknowledgements	7
7. Normative References	7
Appendix A. Example Interoperability Issue	8
Authors' Addresses	8

1. Introduction

[RFC5302] defines the route preferences rules as they apply to Type/Length/Value(TLV)s 128 and 130. [RFC5305] introduced the IP Extended Reachability TLV 135 but did not explicitly adapt the route preference rules defined in [RFC5302] for the new TLV. [RFC5308] defines the IPv6 Reachability TLV 236 and does include an explicit

statement as regards route preference - but the statement introduces use of the up/down bit in advertisements which appear in Level 2 Link State Protocol Data Units (LSPs) which is inconsistent with statements made in [RFC5302] and [RFC5305]. This document defines explicit route preference rules for TLV 135, revises the route preferences rules for TLV 236, and clarifies the usage of the up/down bit when it appears in TLVs in Level 2 LSPs. This document is viewed as a clarification (NOT correction) of [RFC5302] and [RFC5305] and a correction of the route preference rules defined in [RFC5308] to be consistent with the rules for IPv4. It also makes explicit that the same rules apply for the Multi-Topology (MT) equivalent TLVs 235 and 237.

2. Use of the up/down Bit in Level 2 LSPs

The up/down bit was introduced in support of leaking prefixes downwards in the IS-IS level hierarchy. Routes which are leaked downwards have the bit set to 1. Such prefixes MUST NOT be leaked upwards in the hierarchy. So long as we confine ourselves to a single IS-IS instance and the current number of supported levels (two) it is impossible to have a prefix advertised in a Level 2 LSP and have the up/down bit set to 1. However, because [RFC5302] anticipated a future extension to IS-IS which might support additional levels it allowed for the possibility that the up/down bit might be set in a Level-2 LSP and in support of easier migration in the event such an extension was introduced Section 3.3 stated:

"...it is RECOMMENDED that implementations ignore the up/down bit in L2 LSPs, and accept the prefixes in L2 LSPs regardless of whether the up/down bit is set."

[RFC5305] addressed an additional case wherein an implementation included support for multiple virtual routers running IS-IS in different areas. In such a case it is possible to redistribute prefixes between two IS-IS instances in the same manner that prefixes are redistributed from other protocols into IS-IS. This introduced the possibility that a prefix could be redistributed from Level 1 to Level 1 (as well as between Level 2 and Level 2) and in the event the redistributed route was leaked from Level 1 to Level 2 two different routers in different areas would be advertising the same prefix into the Level 2 sub-domain. To prevent this [RFC5305] specified in Section 4.1:

"If a prefix is advertised from one area to another at the same level, then the up/down bit SHALL be set to 1."

However, the statement in [RFC5302] that the up/down bit is ignored in Level 2 LSPs is not altered by [RFC5305].

The conclusion then is that there is no "L2 inter-area route" - and indeed no such route type is defined by [RFC5302]. However, [RFC5308] ignored this fact and introduced such a route type in Section 5 when it specified a preference for "Level 2 down prefix". This is an error which this document corrects.

3. Types of Routes in IS-IS Supported by Extended Reachability TLVs

[RFC5302] is the authoritative reference for the types of routes supported by TLVs 128 and 130. However, a number of attributes supported by those TLVs are NOT supported by TLVs 135, 235, 236, 237. Distinction between internal/external metrics is not supported. In the case of IPv4 TLVs (135 and 235) the distinction between internal and external route types is not supported. It is therefore useful to explicitly state the supported route types for these TLVs.

3.1. Types of Routes Supported by TLVs 135 and 235

This section defines the types of route supported for IPv4 when using TLV 135 [RFC5305] and/or TLV 235 [RFC5120]. The text follows as closely as possible the original text from [RFC5302].

L1 intra-area routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to zero. These IP prefixes are directly connected to the advertising router. These routes are indistinguishable from L1 external routes.

L1 external routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to zero. These IP prefixes are learned from other protocols and are usually not directly connected to the advertising router. These routes are indistinguishable from L1 intra-area routes.

L2 intra-area routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to zero. These IP prefixes are directly connected to the advertising router. These prefixes cannot be distinguished from L1->L2 inter-area routes and/or L2 external routes.

L1->L2 inter-area routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to zero. These IP prefixes are learned via L1 routing and were derived during the L1 Shortest Path First (SPF) computation from prefixes advertised in L1 LSPs in TLV 135 or TLV 235. These prefixes cannot be distinguished from L2 intra-area routes and/or L2 external routes.

L2 external routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to zero. These IP prefixes are

learned from other protocols and are usually not directly connected to the advertising router. These routes are indistinguishable from L2 intra-area routes and/or L1->L2 inter-area routes.

L2->L1 inter-area routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to one. These IP prefixes are learned via L2 routing and were derived during the L2 SPF computation from prefixes advertised in TLV 135 or TLV 235. These routes are indistinguishable from L1->L1 inter-area routes.

L1->L1 inter-area routes: These are advertised in L1 LSPs, in TLV 135 or TLV 235. The up/down bit is set to one. These IP prefixes are learned from another IS-IS instance operating in another area. These routes are indistinguishable from L2->L1 inter-area routes.

L2->L2 inter-area routes: These are advertised in L2 LSPs, in TLV 135 or TLV 235. The up/down bit is set to one but is ignored and treated as if it were set to 0. These IP prefixes are learned from another IS-IS instance operating in another area.

3.2. Types of Routes Supported by TLVs 236 and 237

This section defines the types of route supported for IPv6 when using TLV 236 [RFC5308] and/or TLV 237 [RFC5120].

L1 intra-area routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to zero. The eXternal bit is set to 0. These IPv6 prefixes are directly connected to the advertising router.

L1 external routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to zero. The eXternal bit is set to 1. These IPv6 prefixes are learned from other protocols and are usually not directly connected to the advertising router.

L2 intra-area routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to zero. The eXternal bit is set to 0. These IPv6 prefixes are directly connected to the advertising router. These prefixes cannot be distinguished from L1->L2 inter-area routes.

L1->L2 inter-area routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to zero. The eXternal bit is set to 0. These IPv6 prefixes are learned via L1 routing and were derived during the L1 Shortest Path First (SPF) computation from prefixes advertised in L1 LSPs in TLV 236 or TLV 237. These prefixes cannot be distinguished from L2 intra-area routes.

L2 external routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to zero. the eXternal bit is set to 1. These IPv6 prefixes are learned from other protocols and are usually not directly connected to the advertising router.

L1->L2 external routes: These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to zero. The eXternal bit is set to 1. These IPv6 prefixes are learned via L1 routing and were derived during the L1 Shortest Path First (SPF) computation from L1 external routes advertised in L1 LSPs in TLV 236 or TLV 237. These prefixes cannot be distinguished from L2 external routes.

L2->L1 inter-area routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to one. The eXternal bit is set to 0. These IPv6 prefixes are learned via L2 routing and were derived during the L2 SPF computation from prefixes advertised in TLV 236 or TLV 237. These routes are indistinguishable from L1->L1 inter-area routes.

L2->L1 external routes: These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to one. The eXternal bit is set to 1. These IPv6 prefixes are learned via L2 routing and were derived during the L2 SPF computation from prefixes advertised in TLV 236 or TLV 237.

L1->L1 inter-area routes. These are advertised in L1 LSPs, in TLV 236 or TLV 237. The up/down bit is set to one. The eXternal bit is set to 0. These IP prefixes are learned from another IS-IS instance operating in another area. These routes are indistinguishable from L2->L1 inter-area routes.

L2->L2 inter-area routes. These are advertised in L2 LSPs, in TLV 236 or TLV 237. The up/down bit is set to one but is ignored and treated as if it were set to 0. The eXternal bit is set to 0. These IP prefixes are learned from another IS-IS instance operating in another area.

3.3. Order of Preference for all types of routes supported by TLVs 135 and 235

This document defines the following route preferences for IPv4 routes advertised in TLVs 135 or 235.

1. L1 intra-area routes; L1 external routes
2. L2 intra-area routes; L2 external routes; L1->L2 inter-area routes; L2->L2 inter-area routes

3. L2->L1 inter-area routes; L1->L1 inter-area routes

3.4. Order of Preference for all types of routes supported by TLVs 236 and 237

This document defines the following route preferences for IPv6 routes advertised in TLVs 236 or 237.

1. L1 intra-area routes; L1 external routes

2. L2 intra-area routes; L2 external routes; L1->L2 inter-area routes; L1-L2 external routes; L2-L2 inter-area routes

3. L2->L1 inter-area routes; L2->L1 external routes; L1->L1 inter-area routes

4. IANA Considerations

No IANA actions required.

5. Security Considerations

None.

6. Acknowledgements

TBD

7. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5120] Przygienda, T., Shen, N., and N. Sheth, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", RFC 5120, February 2008.
- [RFC5302] Li, T., Smit, H., and T. Przygienda, "Domain-Wide Prefix Distribution with Two-Level IS-IS", RFC 5302, October 2008.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, October 2008.
- [RFC5308] Hopps, C., "Routing IPv6 with IS-IS", RFC 5308, October 2008.

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Stephane Litkowski
Orange Business Service

Email: stephane.litkowski@orange.com

Stefano Previdi
Cisco Systems
Via Del Serafico 200
Rome 0144
Italy

Email: sprevidi@cisco.com

Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 9, 2014

L. Ginsberg
N. Akiya
Cisco Systems
M. Chen
Huawei
May 08, 2014

Advertising S-BFD Discriminators in IS-IS
draft-ginsberg-isis-sbfd-discriminator-00.txt

Abstract

This document defines a means of advertising one or more S-BFD Discriminators using the IS-IS Router Capability TLV.

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 RFC 2119 [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 November 9, 2014.

Copyright Notice

Copyright (c) 2014 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.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	2
2. Encoding Format	2
3. IANA Considerations	3
4. Security Considerations	3
5. Acknowledgements	3
6. Normative References	4
Authors' Addresses	4

1. Introduction

[S-BFD] defines a simplified mechanism to use Bidirectional Forwarding Detection (BFD)[RFC5880]. This mechanism depends on network nodes knowing the BFD discriminators which each node in the network has reserved for this purpose. Use of the Intermediate System to Intermediate System (IS-IS)[IS-IS] protocol is one possible means of advertising these discriminators.

2. Encoding Format

The IS-IS Router CAPABILITY TLV as defined in [RFC4971] will be used to advertise S-BFD discriminators. A new sub-TLV is defined as described below. S-BFD Discriminators sub-TLVs are formatted as specified in [RFC5305].

	No. of octets
+-----+ Type (to be assigned by IANA - suggested value 19) +-----+	1
+-----+ Length (multiple of 4) +-----+	1
+-----+ Discriminator Value(s) : : +-----+	4/Discriminator

Inclusion of the S-BFD Discriminators sub-TLV in a Router Capability TLV is optional. Multiple S-BFD Discriminators sub-TLVs MAY be advertised by an IS. When multiple S-BFD discriminators are advertised how a given discriminator is mapped to a specific use case is out of scope for this document.

S-BFD discriminator advertisements MAY be flooded within an area or throughout the domain using the procedures specified in [RFC4971].

3. IANA Considerations

This document requires the definition of a new sub-TLV in the Sub-TLVs for TLV 242 registry. The value written below is a suggested value subject to assignment by IANA.

Value	Description
-----	-----
19	S-BFD Discriminators

4. Security Considerations

Security concerns for IS-IS are addressed in [IS-IS], [RFC5304], and [RFC5310]. Introduction of the S-BFD Discriminators sub-TLV introduces no new security risks for IS-IS.

Advertisement of the S-BFD discriminators does make it possible for attackers to initiate S-BFD sessions using the advertised information. The vulnerabilities this poses and how to mitigate them are discussed in the Security Considerations section of [S-BFD].

5. Acknowledgements

The authors wish to thank Sam Aldrin, Manav Bhatia, and Carlos Pignataro for input essential to defining the needed functionality.

6. Normative References

- [IS-IS] "Intermediate system to Intermediate system intra-domain routeing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode Network Service (ISO 8473), ISO/IEC 10589:2002, Second Edition.", Nov 2002.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4971] Vasseur, JP., Shen, N., and R. Aggarwal, "Intermediate System to Intermediate System (IS-IS) Extensions for Advertising Router Information", RFC 4971, July 2007.
- [RFC5304] Li, T. and R. Atkinson, "IS-IS Cryptographic Authentication", RFC 5304, October 2008.
- [RFC5305] Li, T. and H. Smit, "IS-IS Extensions for Traffic Engineering", RFC 5305, October 2008.
- [RFC5310] Bhatia, M., Manral, V., Li, T., Atkinson, R., White, R., and M. Fanto, "IS-IS Generic Cryptographic Authentication", RFC 5310, February 2009.
- [RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, June 2010.
- [S-BFD] "Seamless Bidirectional Forwarding Detection (S-BFD), draft-akiya-bfd-seamless-base-03(work in progress)", April 2014.

Authors' Addresses

Les Ginsberg
Cisco Systems
510 McCarthy Blvd.
Milpitas, CA 95035
USA

Email: ginsberg@cisco.com

Nobo Akiya
Cisco Systems

Email: nobo@cisco.com

Mach(Guoyi) Chen
Huawei

Email: mach.chen@huawei.com

ISIS Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 29, 2014

S. Litkowski
Orange
June 27, 2014

Yang Data Model for ISIS protocol
draft-litkowski-isis-yang-isis-cfg-01

Abstract

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

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 December 29, 2014.

Copyright Notice

Copyright (c) 2014 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	5
2.3. Per level parameters	5
2.4. Per interface parameters	6
2.5. Operational states	7
3. RPC operations	8
4. Notifications	8
5. Interaction with other YANG modules	9
6. Yang module	9
7. Security Considerations	45
8. Acknowledgements	46
9. IANA Considerations	46
10. Normative References	46
Appendix A. Example: NETCONF <get> Reply	47
Author's Address	53

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 Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- o Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- 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 containers :

- o isis-cfg : that contains writable configuration objects.
- o isis-state : that contains read-only states.

The container isis-cfg and isis-state are augmenting the "routing-protocol" lists in ietf-routing module with specific ISIS parameters.

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

```

module: isis
augment /rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route:
  +--ro metric?          uint32
  +--ro tag*             uint32
  +--ro route-type?     enumeration

augment /rt:active-route/rt:output/rt:route:
  +--ro metric?          uint32
  +--ro tag*             uint32
  +--ro route-type?     enumeration

augment
  /rt:routing/rt:routing-instance/rt:routing-protocols/rt:routing-protocol:
    +--rw isis
      +--rw isis-level?          isis-level
      +--rw nsap-address         simple-iso-address
      +--rw ipv4-router-id?     inet:ipv4-address
      +--rw ipv6-router-id?     inet:ipv6-address
      +--rw reference-bandwidth? uint32
      +--rw lsp-mtu?            uint16
      +--rw lsp-lifetime?       uint16
      +--rw lsp-refresh?        uint16
      +--rw psnp-authentication? boolean
      +--rw csnp-authentication? boolean
      +--rw hello-authentication? boolean
      +--rw authentication-key? string
      +--rw authentication-type? enumeration
      +--rw isis-multi-topology-cfg
      |   ...
      +--rw isis-level-1-cfg
      |   ...

```

```

    +--rw isis-level-2-cfg
    |   ...
    +--rw overload
    |   +--rw status?      boolean
    |   +--rw timeout?     uint16
    +--rw interfaces
    |   +--rw interface* [name]
    |   ...

augment
/rt:routing-state/rt:routing-instance/rt:routing-protocols/rt:routing-protocol:
+--ro isis-state
+--ro adjacencies
|   +--ro adjacency* [interface]
|   |   +--ro interface      string
|   |   +--ro level?        isis-level
|   |   +--ro state?        enumeration
+--ro spf-log
|   +--ro event* [id]
|   |   +--ro id              uint32
|   |   +--ro spf-type?      enumeration
|   |   +--ro level?        isis-level
|   |   +--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        isis-lsp-id
|   |   |   +--ro sequence?  uint32
+--ro lsp-log
|   +--ro event* [id]
|   |   +--ro id              uint32
|   |   +--ro level?        isis-level
|   |   +--ro lsp
|   |   |   +--ro lsp?        isis-lsp-id
|   |   |   +--ro sequence?  uint32
|   |   +--ro received-timestamp? yang:timestamp
+--ro database
|   +--ro level-1
|   |   ...
|   +--ro level-2
|   |   ...
+--ro hostnames
|   +--ro hostname* [system-id]
|   |   +--ro system-id      isis-system-id
|   |   +--ro hostname?     string

```

2.1. ISIS configuration

The ISIS configuration container is divided in :

- o Global parameters.
- o Level specific parameters (see Section 2.3).
- o Per interface configuration (see Section 2.4).

2.2. Multitopology parameters

The multitopology section is used to enable support of MT extensions for specific address families.

A boolean is associated with each topology type, defining if the topology is enabled or not.

```

  +--rw isis-multi-topology-cfg
  |   +--rw ipv4-unicast?      boolean
  |   +--rw ipv6-unicast?      boolean
  |   +--rw ipv4-multicast?    boolean
  |   +--rw ipv6-multicast?    boolean
```

2.3. Per level parameters

The level parameter section of the ISIS instance describes the global parameters for level 1 and 2 including authentication, protocol preferences, default metrics ...

Level specific parameters override parameters contained directly under isis-cfg container.

```

+--rw isis-level-1-cfg
|   +--rw enabled?                boolean
|   +--rw psnp-authentication?    boolean
|   +--rw csnp-authentication?    boolean
|   +--rw hello-authentication?   boolean
|   +--rw authentication-key?     string
|   +--rw authentication-type?    enumeration
|   +--rw metric-type?            enumeration
|   +--rw preference?             uint8
|   +--rw external-preference?    uint8
|   +--rw default-ipv4-unicast-metric? isis-wide-metric
|   +--rw default-ipv6-unicast-metric? isis-wide-metric
|   +--rw default-ipv4-multicast-metric? isis-wide-metric
|   +--rw default-ipv6-multicast-metric? isis-wide-metric
+--rw isis-level-2-cfg
|   +--rw enabled?                boolean
|   +--rw psnp-authentication?    boolean
|   +--rw csnp-authentication?    boolean
|   +--rw hello-authentication?   boolean
|   +--rw authentication-key?     string
|   +--rw authentication-type?    enumeration
|   +--rw metric-type?            enumeration
|   +--rw preference?             uint8
|   +--rw external-preference?    uint8
|   +--rw default-ipv4-unicast-metric? isis-wide-metric
|   +--rw default-ipv6-unicast-metric? isis-wide-metric
|   +--rw default-ipv4-multicast-metric? isis-wide-metric
|   +--rw default-ipv6-multicast-metric? isis-wide-metric

```

2.4. Per interface parameters

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

Each interface has interface-specific parameters and level-specific parameters. A level-specific parameter always override interface-specific parameter and an interface-specific parameter always override an isis global parameter (defined in isis-cfg).

```

+--rw interfaces
|   +--rw interface* [name]
|       +--rw name                leafref
|       +--rw level?              isis-level
|       +--rw lsp-interval?       uint16
|       +--rw passive?            boolean
|       +--rw csnp-interval?      uint16
|       +--rw hello-authentication-type? enumeration
|       +--rw hello-authentication-key? string

```

```

+--rw hello-interval?          uint16
+--rw hello-multiplier?        uint16
+--rw hello-padding?           boolean
+--rw ipv4-unicast?             boolean
+--rw ipv6-unicast?             boolean
+--rw ipv4-multicast?           boolean
+--rw ipv6-multicast?           boolean
+--rw interface-type?           enumeration
+--rw enabled?                  boolean
+--rw tag*                      uint32
+--rw level-1
|   +--rw hello-authentication-type? enumeration
|   +--rw hello-authentication-key?  string
|   +--rw hello-interval?            uint16
|   +--rw hello-multiplier?          uint16
|   +--rw ipv4-unicast?               boolean
|   +--rw ipv6-unicast?               boolean
|   +--rw ipv4-multicast?             boolean
|   +--rw ipv6-multicast?             boolean
|   +--rw priority?                  uint8
|   +--rw ipv4-unicast-metric?        isis-wide-metric
|   +--rw ipv6-unicast-metric?        isis-wide-metric
|   +--rw ipv4-multicast-metric?      isis-wide-metric
|   +--rw ipv6-multicast-metric?      isis-wide-metric
|   +--rw passive?                   boolean
+--rw level-2
|   +--rw hello-authentication-type? enumeration
|   +--rw hello-authentication-key?  string
|   +--rw hello-interval?            uint16
|   +--rw hello-multiplier?          uint16
|   +--rw ipv4-unicast?               boolean
|   +--rw ipv6-unicast?               boolean
|   +--rw ipv4-multicast?             boolean
|   +--rw ipv6-multicast?             boolean
|   +--rw priority?                  uint8
|   +--rw ipv4-unicast-metric?        isis-wide-metric
|   +--rw ipv6-unicast-metric?        isis-wide-metric
|   +--rw ipv4-multicast-metric?      isis-wide-metric
|   +--rw ipv6-multicast-metric?      isis-wide-metric
|   +--rw passive?                   boolean

```

2.5. Operational states

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

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

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

4. Notifications

The "ietf-isis" module a new notification "isis-adjacency-updown". This notification is triggered when an ISIS adjacency moves to Up or Down state.

This notification provides details on the affected adjacency and its new state.

notifications:

```
+---n isis-adjacency-updown
  +---ro interface?          string
  +---ro neighbor?           string
  +---ro neighbor-system-id? isis-system-id
  +---ro isis-level?         isis-level
  +---ro state?              enumeration
  +---ro reason?             string
```

5. Interaction with other YANG modules

The "isis-cfg" 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-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@2014-06-25.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;
  }

  organization
    "IETF ISIS Working Group";
```

```
contact
  "WG List:    <mailto:isis-wg@ietf.org>

  Editor:      Stephane Litkowski
                <mailto:stephane.litkowski@orange.com>";

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

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-state {
        description "Identity for the ISIS routing protocol
        operational states.";
    }

    identity isis-adjacency-updown {
        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.";
    }

    typedef isis-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 leafs that reference state data of
            an ISIS protocol instance.";
    }

    typedef isis-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-1-2" {
            description
                "This enum describes both level capability.";
        }
    }
    description
        "This type defines ISIS level of an object.";
}

typedef isis-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 simple-iso-address {
    type string {
        pattern '[0-9A-Fa-f]{2}\.([0-9A-Fa-f]{4}\.){0,3}'
            +'[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.[0-9A-Fa-f]{4}\.'
            +'[0-9][0-9]';
    }
    description
        "This type defines simple iso address format,
        it looks like : area_id.systemid.nsel
        The area ID is at least 1 byte of AFI, and is up to
        13 bytes.";
}

typedef isis-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 isis-wide-metric {
    type uint32 {
        range "0 .. 16777215";
    }
    description
```

```
        "This type defines wide style format
        of ISIS metric.";
    }

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

    grouping isis-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 uint32;
            description
                "This leaf describes list of tags associated
                with the route.";
        }
        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.";
}

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

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

grouping isis-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 isis-std-metric;
        description
            "This leaf describes the isis default metric value";
    }
    container delay-metric {
        leaf metric {
            type isis-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 isis-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 expense metric.";
    }
}
```

```
    container error-metric {
      leaf metric {
        type isis-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 error metric.";
    }
  }

  grouping isis-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 isis-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.";
```

```
    }  
  }  
  
  grouping isis-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 isis-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.";  
    }  
  }  
  
  grouping isis-neighbor-extended {  
    description  
      "This group defines attributes of an  
      ISIS extended neighbor.";  
    leaf neighbor-id {  
      type isis-system-id;  
      description  
        "This leaf describes the system-id of the neighbor.";  
    }  
    leaf metric {  
      type isis-wide-metric;  
      description
```

```
        "This leaf describes the isis metric value";
    }
}

grouping isis-neighbor {
    description
        "This group defines attributes of an
        ISIS standard neighbor.";
    leaf neighbor-id {
        type isis-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 isis-std-metric;
        description
            "This leaf describes the isis default metric value";
    }
    container delay-metric {
        leaf metric {
            type isis-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 isis-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 expense metric.";
}
container error-metric {
    leaf metric {
        type isis-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 error metric.";
}
}

grouping isis-database {
    description
        "This group defines attributes of an
        ISIS database (Link State DB).";
    leaf lsp-id {
        type isis-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 refered metric.";
    }
    bit ATTACHED-EXPENSE {
      description
        "If set, the originator is attached to
        another area using the refered metric.";
    }
    bit ATTACHED-DELAY {
      description
        "If set, the originator is attached to
        another area using the refered metric.";
    }
    bit ATTACHED-DEFAULT {
      description
        "If set, the originator is attached to
        another area using the refered 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 isis-neighbor-extended;
    description
      "List of neighbors.";
  }
  description
    "This leaf describes list of ISIS extended neighbors.
    ISIS reference is TLV 2.";
}
```

```
container authentication {
  leaf authentication-type {
    type uint8;
    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 isis-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 isis-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 isis-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 isis-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 isis-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 refered 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 isis-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 isis-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 isis-prefix-ipv6-extended;
    description
      "List of prefixes.";
  }
  description
    "This container describes list of ipv6
    reachability information.
    ISIS reference is TLV 236.";
```

```
    }

    container router-capabilities {
        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.";
    }
}

grouping isis-address-family-cfg {
    description
        "This group defines address-family-cfg
        global configuration for ISIS.";
    leaf ipv4-unicast {
        type boolean;
        description
            "This leaf defines if IPv4 unicast is activated.";
    }
    leaf ipv6-unicast {
        type boolean;
        description
            "This leaf defines if IPv6 unicast is activated.";
    }
    leaf ipv4-multicast {
        type boolean;
        description
            "This leaf defines if IPv4 multicast is activated.";
    }
    leaf ipv6-multicast {
        type boolean;
        description
            "This leaf defines if IPv6 multicast is activated.";
    }
}

grouping isis-interface-hello-cfg {
    description
        "This group defines hello interface
        parameters for ISIS.";
    leaf hello-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 leaf describes the authentication type
      to be used in hello messages.";
  }
  leaf hello-authentication-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 hello-interval {
    type uint16;
    units "seconds";
    description
      "This leaf defines the interval of hello messages.";
  }
  leaf hello-multiplier {
    type uint16;
    description
      "This leaf defines the number of hello failed to be
      received before declaring the adjacency down.";
  }
}

grouping isis-interface-level-cfg {
  description
    "This group defines level specific
    configuration for ISIS interfaces.";
  uses isis-interface-hello-cfg;

  uses isis-address-family-cfg;
```

```
    leaf priority {
      type uint8 {
        range "0 .. 127";
      }

      description
        "This leaf describes the priority of the interface
        for DIS election.";
    }
    leaf ipv4-unicast-metric {
      type isis-wide-metric;
      description
        "This leaf describes the IPv4 unicast metric
        of the interface.";
    }
    leaf ipv6-unicast-metric {
      type isis-wide-metric;
      description
        "This leaf describes the IPv6 unicast metric
        of the interface.";
    }
    leaf ipv4-multicast-metric {
      type isis-wide-metric;
      description
        "This leaf describes the IPv4 multicast metric
        of the interface.";
    }
    leaf ipv6-multicast-metric {
      type isis-wide-metric;
      description
        "This leaf describes the IPv6 multicast metric
        of the interface.";
    }
    leaf passive {
      type boolean;
      default "false";
      description
        "This leaf defines if interface is in passive mode
        (ISIS not running, but network is advertised).";
    }
  }
}

grouping isis-authentication-cfg {
  description
    "This group defines authentication
    configuration for ISIS.";
  leaf psnp-authentication {
    type boolean;
  }
}
```

```
        default "true";
        description
            "This leaf describes if PSNP messages must be
            authenticated.";
    }
    leaf csnp-authentication {
        type boolean;
        default "true";
        description
            "This leaf describes if CSNP messages must be
            authenticated.";
    }
    leaf hello-authentication {
        type boolean;
        default "true";
        description
            "This leaf describes if HELLO messages must be
            authenticated.";
    }
    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 upon a get-config reply.
            Authors recommends
            to use MD5 hash to present the authentication-key";
    }
    leaf 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 leaf describes the authentication type
            to be used.";
    }
}
```

```
grouping isis-level-cfg {
  description
    "This group defines level specific
    global configuration for ISIS.";
  leaf enabled {
    type boolean;
    default "true";
    description
      "This leaf defines the status of the administrative
      status of the level (Active / not active).";
  }

  uses isis-authentication-cfg;

  leaf metric-type {
    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.";
  }
  leaf preference {
    type uint8;
    description
      "This leaf defines the protocol preference.";
  }
  leaf external-preference {
    type uint8;
    description
      "This leaf defines the protocol preference for external
      routes.";
  }
  leaf default-ipv4-unicast-metric {
    type isis-wide-metric;
    description
```

```
        "This leaf defines the IPv4 unicast default metric.";
    }
    leaf default-ipv6-unicast-metric {
        type isis-wide-metric;
        description
            "This leaf defines the IPv6 unicast default metric.";
    }
    leaf default-ipv4-multicast-metric {
        type isis-wide-metric;
        description
            "This leaf defines the IPv4 multicast default metric.";
    }
    leaf default-ipv6-multicast-metric {
        type isis-wide-metric;
        description
            "This leaf defines the IPv6 multicast default metric.";
    }
}

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 {
        leaf isis-level {
            type isis-level;
            default "level-1-2";
            description
                "This leaf describes the type of ISIS node.
                A node can be level-1-only, level-2-only
                or level-1-2.
                ";
        }
        leaf nsap-address {
            type simple-iso-address;
            description
                "This leaf defines the NSAP address of the node
                in a simple format. This parameter is mandatory.";
        }
        leaf ipv4-router-id {
            type inet:ipv4-address;
            description
```

```
        "Router ID value that would be used in TLV134.";
    }
    leaf ipv6-router-id {
        type inet:ipv6-address;
        description
            "Router ID value that would be used in TLV234.";
    }
    leaf 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 {
        type uint16;
        units "seconds";
        description
            "This leaf describes the refresh interval of the
            router LSP in seconds.";
    }
}

uses isis-authentication-cfg;

container isis-multi-topology-cfg {
    uses isis-address-family-cfg;
    description
        "This container describes activation of MT extensions
        for supporting new address families.";
}

container isis-level-1-cfg {
```

```
    uses isis-level-cfg;
    description
        "Defines configuration parameters of level 1.";
}

container isis-level-2-cfg {
    uses isis-level-cfg;
    description
        "Defines configuration parameters of level 2.";
}

container overload {
    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.";
    }
    description
        "This leaf describes if the router is
        set to overload state.";
}

container interfaces {
    list interface {
        key "name";
        leaf name {
            type leafref {
                path "/rt:routing/rt:routing-instance/"
                    +"rt:interfaces/rt:interface/rt:name";
            }
            description
                "Reference to the interface within
                the routing-instance.";
        }
        leaf level {
            type isis-level;
            default "level-1-2";
            description
                "This leaf defines the associated ISIS level
                of the interface.";
        }
    }
}
```

```
}
leaf lsp-interval {
    type uint16;
    units "milliseconds";
    description
        "This leaf defines the interval between LSP
        transmissions in msec";
}
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.";
}

uses isis-interface-hello-cfg;

leaf hello-padding {
    type boolean;
    description
        "This leaf defines if ISIS Hellos would be
        padded up to MTU size.";
}

uses isis-address-family-cfg;

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

    container level-1 {
        uses isis-interface-level-cfg;
        description
            "This container defines the level 1 specific
            configuration of the interface.";
    }

    container level-2 {
        uses isis-interface-level-cfg;
        description
            "This container defines the level 2 specific
            configuration of the interface.";
    }

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

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-state {
  config false;
  container adjacencies {

    list adjacency {
      key interface;

      leaf interface {
        type string;
        description
          "This leaf describes the name
          of the interface.";
      }
      leaf level {
        type uint8 {
          range "1 .. 2";
        }
        description
          "This leaf describes the associated
          ISIS level of the interface.
          The value of the level can only be 1
          or 2.";
      }
    }
    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.";
    }
}
description
    "This leaf describes the state of the
    interface.";
}
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 uint8 {
    range "1 .. 2";
  }
  description
    "This leaf describes the level affected bytes
    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 isis-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 uint8 {
                range "1 .. 2";
            }
            description
                "This leaf describes the level affected bytes
                the computation.";
        }
    }
    container lsp {
        leaf lsp {

            type isis-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.";
    }

    description
        "List of LSP events.";
}

description
    "This container lists the LSP reception events.
    Local LSP modification are also contained in the
    list.";
}
container database {
    container level-1 {
        list lsp {
            key lsp-id;

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

        description
            "This container describes the list of LSPs
            in the level-1 database.";
    }
    container level-2 {
        list lsp {
            key lsp-id;

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

        description
            "This container describes the list of LSPs
            in the level-2 database.";
    }
}
description
    "This container describes ISIS Link State
```

```
        databases.";
    }
    container hostnames {

        list hostname {
            key system-id;
            leaf system-id {
                type isis-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
    "This container defines various ISIS states objects.";
}

/* RPC methods */

rpc clear-isis-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 isis-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 isis-level {
  type isis-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-isis-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 isis-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 isis-level {
        type isis-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 isis-adjacency-updown {
    leaf interface {
        type string;
        description
            "Describes the interface of the adjacency";
    }
    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 isis-system-id;
        description
            "Describes the system-id of the neighbor.";
    }
    leaf isis-level {
        type isis-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.";
}

}

<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 :

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

- o 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.
- o 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. Acknowledgements

9. IANA Considerations

10. Normative References

- [I-D.ietf-netmod-routing-cfg]
Lhotka, L., "A YANG Data Model for Routing Management", draft-ietf-netmod-routing-cfg-15 (work in progress), May 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"
  xmlns:isis="urn:ietf:params:xml:ns:yang:ietf-isis"
  xmlns:v4ur="urn:ietf:params:xml:ns:yang:ietf-ipv4-unicast-routing"
  xmlns:rt="urn:ietf:params:xml:ns:yang:ietf-routing"
  xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
  xmlns:ip="urn:ietf:params:xml:ns:yang:ietf-ip">
  <if:interfaces>
    <if:interface>
      <if:name>eth0</if:name>
      <ip:ipv4>
        <ip:address>
          <ip:ip>10.0.0.1</ip:ip>
          <ip:prefix-length>30</ip:prefix-length>
        </ip:address>
      </ip:ipv4>
    </if:interface>
    <if:interface>
      <if:name>lo0</if:name>
      <ip:ipv4>
        <ip:address>
          <ip:ip>192.168.0.2</ip:ip>
          <ip:prefix-length>32</ip:prefix-length>
        </ip:address>
      </ip:ipv4>
    </if:interface>
  </if:interfaces>
  <if:interfaces-state>
    <if:interface>
      <if:name>eth0</if:name>
      <if:oper-status>up</if:oper-status>
      <if:statistics>
        <if:discontinuity-time>
          2014-12-02T12:14:25.123Z
        </if:discontinuity-time>
        <if:in-octets>100</if:in-octets>
        <if:in-unicast-pkts>10</if:in-unicast-pkts>
        <if:in-broadcast-pkts>0</if:in-broadcast-pkts>
      </if:statistics>
    </if:interface>
  </if:interfaces-state>
</data>
```

```
<if:in-multicast-pkts>0</if:in-multicast-pkts>
<if:in-discards>0</if:in-discards>
<if:in-errors>0</if:in-errors>
<if:in-unknown-protos>0</if:in-unknown-protos>
<if:out-octets>541</if:out-octets>
<if:out-unicast-pkts>100</if:out-unicast-pkts>
<if:out-broadcast-pkts>0</if:out-broadcast-pkts>
<if:out-multicast-pkts>0</if:out-multicast-pkts>
<if:out-discards>0</if:out-discards>
<if:out-errors>0</if:out-errors>
</if:statistics>
<ip:ipv4>
  <ip:address>
    <ip:ip>10.0.0.1</ip:ip>
    <ip:prefix-length>30</ip:prefix-length>
  </ip:address>
</ip:ipv4>
</if:interface>
<if:interface>
  <if:name>lo0</if:name>
  <if:oper-status>up</if:oper-status>
  <if:statistics>
    <if:discontinuity-time>
      2014-12-02T12:14:25.123Z
    </if:discontinuity-time>
    <if:in-octets>100</if:in-octets>
    <if:in-unicast-pkts>10</if:in-unicast-pkts>
    <if:in-broadcast-pkts>0</if:in-broadcast-pkts>
    <if:in-multicast-pkts>0</if:in-multicast-pkts>
    <if:in-discards>0</if:in-discards>
    <if:in-errors>0</if:in-errors>
    <if:in-unknown-protos>0</if:in-unknown-protos>
    <if:out-octets>541</if:out-octets>
    <if:out-unicast-pkts>100</if:out-unicast-pkts>
    <if:out-broadcast-pkts>0</if:out-broadcast-pkts>
    <if:out-multicast-pkts>0</if:out-multicast-pkts>
    <if:out-discards>0</if:out-discards>
    <if:out-errors>0</if:out-errors>
  </if:statistics>
  <ip:ipv4>
    <ip:address>
      <ip:ip>192.168.0.2</ip:ip>
      <ip:prefix-length>32</ip:prefix-length>
    </ip:address>
  </ip:ipv4>
</if:interface>
</if:interfaces-state>
<rt:routing-state>
```

```
<rt:routing-instance>
  <rt:name>rtrl</rt:name>
  <rt:id>1</rt:id>
  <rt:router-id>192.0.2.1</rt:router-id>
  <rt:interfaces>
    <rt:interface>
      <rt:name>eth0</rt:name>
    </rt:interface>
  </rt:interfaces>
  <rt:routing-protocols>
    <rt:routing-protocol>
      <rt:name>ISIS1</rt:name>
      <rt:type>isis:isis</rt:type>
      <isis:isis-state>
        <isis:adjacencies>
          <isis:adjacency>
            <isis:interface>eth0</isis:interface>
            <isis:level>2</isis:level>
            <isis:state>Up</isis:state>
          </isis:adjacency>
        </isis:adjacencies>
        <isis:spf-log>
          <isis:event>
            <isis:id>18979</isis:id>
            <isis:spf-type>full</isis:spf-type>
            <isis:level>2</isis:level>
            <isis:spf-delay>150</isis:spf-delay>
            <isis:schedule-timestamp>
              1403612245
            </isis:schedule-timestamp>
            <isis:start-timestamp>
              1403612399
            </isis:start-timestamp>
            <isis:end-timestamp>
              1403612420
            </isis:end-timestamp>
            <isis:trigger-lsp>
              <isis:lsp>0000.1245.1245.01-01</isis:lsp>
              <isis:sequence>125458</isis:sequence>
            </isis:trigger-lsp>
          </isis:event>
        </isis:spf-log>
        <isis:lsp-log>
          <isis:event>
            <isis:id>1245</isis:id>
            <isis:level>2</isis:level>
            <isis:lsp>
              <isis:lsp>0000.1245.1245.01-01</isis:lsp>
```

```
<isis:sequence>125458</isis:sequence>
</isis:lsp>
<isis:received-timestamp>
1403612420
</isis:received-timestamp>
</isis:event>
</isis:lsp-log>
<isis:database>
<isis:level-2>
<isis:lsp>
<isis:lsp-id>0000.1245.1245.01-01</isis:lsp-id>
<isis:checksum>1245</isis:checksum>
<isis:remaining-lifetime>
45
</isis:remaining-lifetime>
<isis:sequence>125458</isis:sequence>
<isis:extended-is-neighbor>
<isis:neighbor>
<isis:neighbor-id>
0000.1245.9999.00
</isis:neighbor-id>
<isis:metric>100</isis:metric>
</isis:neighbor>
</isis:extended-is-neighbor>

<isis:protocol-supported>
204
</isis:protocol-supported>

<isis:ipv4-addresses>
192.168.0.2
</isis:ipv4-addresses>
<isis:extended-ipv4-reachability>
<isis:prefixes>
<isis:ip-prefix>192.168.0.2</isis:ip-prefix>
<isis:up-down>false</isis:up-down>
<isis:prefix-len>32</isis:prefix-len>
<isis:metric>30490</isis:metric>
<isis:tag>200</isis:tag>
</isis:prefixes>
</isis:extended-ipv4-reachability>
<isis:dynamic-hostname>
rtrl
</isis:dynamic-hostname>

</isis:lsp>
</isis:level-2>
</isis:database>
```

```
<isis:hostnames>
  <isis:hostname>
    <isis:system-id>
      0000.1245.9999.00
    </isis:system-id>
    <isis:hostname>rtrl</isis:hostname>
  </isis:hostname>
</isis:hostnames>
</isis:isis-state>
</rt:routing-protocol>
</rt:routing-protocols>
</rt:routing-instance>
<rt:ribs>
  <rt:rib>
    <rt:name>ipv4-master</rt:name>
    <rt:id>1</rt:id>
    <rt:address-family>
      v4ur:ipv4-unicast
    </rt:address-family>
    <rt:routes>
      <rt:route>
        <rt:id>124554657</rt:id>
        <rt:outgoing-interface>eth0</rt:outgoing-interface>
        <rt:source-protocol>isis:isis</rt:source-protocol>
        <rt:last-updated>
          2013-07-02T18:02:45+01:00
        </rt:last-updated>
        <v4ur:destination-prefix>
          10.0.0.0/24
        </v4ur:destination-prefix>
        <isis:metric>750</isis:metric>
        <isis:route-type>l2-up-internal</isis:route-type>

      </rt:route>
    </rt:routes>

  </rt:rib>
</rt:ribs>

</rt:routing-state>
<rt:routing>
  <rt:routing-instance>
    <rt:name>rtrl</rt:name>
    <rt:router-id>192.0.2.1</rt:router-id>
    <rt:interfaces>
      <rt:interface>
        <rt:name>eth0</rt:name>
      </rt:interface>
```

```
<rt:interface>
  <rt:name>lo0</rt:name>
</rt:interface>
</rt:interfaces>
<rt:routing-protocols>
  <rt:routing-protocol>
    <rt:name>ISIS1</rt:name>
    <rt:type>isis:isis</rt:type>
    <isis:isis>
      <isis:isis-level>level-2</isis:isis-level>
      <isis:nsap-address>
        49.0002.0000.1245.9999.00
      </isis:nsap-address>
      <isis:lsp-lifetime>65535</isis:lsp-lifetime>
      <isis:lsp-refresh>65000</isis:lsp-refresh>
      <isis:authentication-key>
        f567acafcd6578861df5683268$
      </isis:authentication-key>
      <isis:authentication-type>
        plaintext
      </isis:authentication-type>
      <isis:isis-multi-topology-cfg>
        <isis:ipv4-unicast>true</isis:ipv4-unicast>
      </isis:isis-multi-topology-cfg>

      <isis:isis-level-2-cfg>
        <isis:metric-type>wide-only</isis:metric-type>
        <isis:default-ipv4-unicast-metric>
          11111111
        </isis:default-ipv4-unicast-metric>
      </isis:isis-level-2-cfg>
    <isis:interfaces>
      <isis:interface>
        <isis:name>eth0</isis:name>
        <isis:lsp-interval>100</isis:lsp-interval>
        <isis:csnp-interval>10</isis:csnp-interval>
        <isis:interface-type>
          point-to-point
        </isis:interface-type>
        <isis:level-2>
          <isis:ipv4-unicast-metric>
            200
          </isis:ipv4-unicast-metric>
        </isis:level-2>
      </isis:interface>
      <isis:interface>
        <isis:name>lo0</isis:name>
        <isis:level-2>
```

```
<isis:ipv4-unicast-metric>
1
</isis:ipv4-unicast-metric>
</isis:level-2>
<isis:passive>true</isis:passive>
</isis:interface>
</isis:interfaces>
</isis:isis>
</rt:routing-protocol>
</rt:routing-protocols>
</rt:routing-instance>
</rt:routing>
</data>
```

Author's Address

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Network Working Group
Internet Draft

Category: Standard Track

L. Yong
W. Hao
D. Eastlake
Huawei
A. Qu
J. Hudson
Brocade

Expires: December 2014

June 12, 2014

IS-IS Protocol Extension For Building Distribution Trees
draft-yong-isis-ext-4-distribution-tree-02

Abstract

This document proposes an IS-IS protocol extension for automatically building bi-directional distribution trees to transport multi-destination traffic in an IP network.

Status of this document

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on December 12, 2014.

Copyright Notice

Copyright (c) 2013 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.

Table of Contents

1. Introduction.....	3
1.1. Conventions used in this document.....	4
2. IS-IS Protocol Extension.....	5
2.1. RTADDR sub-TLV.....	5
2.2. RTADDRV6 sub-TLV.....	6
2.3. The Group Address Sub-TLV.....	7
3. Procedures.....	8
3.1. Distribution Tree Computation.....	8
3.2. Parent Selection.....	8
3.3. Parallel Local Link Selection.....	9
3.4. Tree Selection for a Group.....	10
3.5. Pruning a Distribution Tree for a Group.....	10
3.6. Reverse Path Forwarding Check (RPFC).....	10
3.7. Forwarding Using a Pruned Distribution Tree.....	11
3.8. Local Forwarding at Edge Router.....	11
3.9. Distribution Tree across different IGP Levels.....	12
4. Mobility Support.....	14
4.1. Listener moves from one edge router to another.....	14
4.2. Source host moves from one edge router to another.....	14
5. Backward Compatibility.....	14
6. Interworking with PIM.....	14
7. Security Considerations.....	14
8. IANA Considerations.....	14
9. Acknowledgements.....	15
10. References.....	15
10.1. Normative References.....	15
10.2. Informative References.....	15

1. Introduction

Computer virtualization and cloud applications motivate the DC network virtualization technology [NVO3FRWK]. This technology decouples the end-points networking from the DC physical infrastructure network in terms of address space and configuration [NVO3FRWK].

DC network virtualization solutions are required to carry all types of traffic in today's DC physical networks including multi-destination traffic. It is also desirable to use an IP network as the DC underlying network for the overlay virtual networks [NVO3FRWK].

IP network technology does not yet support multi-destination traffic forwarding. A variety of Protocol Independent Multicast (PIM) solutions [RFC4601] [RFC5015] are designed to carry IP multicast traffic over IP networks. However DC infrastructure for multi-tenancy application is simple IGP domain where using PIM for multicast transport has several drawbacks. This is because the PIM use their own hello protocol and hop-to-hop Join/Leave message so each router does not have global information about the receivers; in the PIM, the data packets could be forwarded unnecessarily to the Rendezvous Point(RP), and then get dropped there when no receiver at all or the sender and receivers for a multicast group are on the same branch towards the RP. This can unnecessarily consume network resources. Furthermore PIM solutions maintain a lot of soft-state, have intensive CPU utilization, and have additional convergence time, besides the IGP's, under a failure condition.

Although the PIM protocol is mature and has been deployed in IP networks, applying PIM to DC IP network that supports the Network Virtualization Overlays can be an extremely challenging [MCASTISS] [DCMCAST]. For example, VXLAN [VXLAN] solutions require multicast support in the underlying network to simulate overlay L2 broadcast capability, where every edge node in an overlay virtual network (VN) is a multicast source and receiver. An overlay VN topology may be sparse and dynamic compared to the underlying IP network topology. Also a large number of overlay VNs may exist in a DC, which PIM solutions can't scale to.

Furthermore IP Overlay based network virtualization technology has been adopted by network vendors to create a VN automatically, self-healing, multi-service fabric to achieve the goal of a SDN capable fabric which is open, programmable, and elastic. Within the fabric, it is a closed IP network carrying all types of traffic, hence

having one control plane protocol to support both uni-destination and multi-destination forwarding.

This is the motivation to extend IGP protocol in support multicast transport so one IGP protocol can support both unicast and multicast transport. This document uses extensions to the IS-IS protocol to build a distribution tree for multi-destination traffic transport in an IP network. A router uses either a Router Capabilities TLV or an MT Router Capabilities TLV to announce the tree root address and the multicast groups associated to the tree. With this information, routers in the IGP can compute rooted distribution trees by using the link state information, i.e. LSDB, and shortest path algorithm. Edge routers include information in their LSPs to announce their multicast group-memberships. Routers perform distribution tree pruning for each multicast group based on other router's group membership announcements. A router forwards the multi-destination traffic along the pruned tree.

In case that edge router needs to get the host membership of a multicast group, edge routers may use IGMP query messages [RFC3376] to inform the attached hosts and the hosts use IGMP report message to response with their interested multicast group(s).

In cases where the solution described in this document applies to the underlying network that transports overlay virtual networks [NVO3FRWK], mapping between an overlay multicast group and a underlying multicast group is necessary. Edge routers further need to perform packet encapsulation/decapsulation.[NVO3FRWK]

The benefits of this solution are 1) protocol convergence: use single protocol for both unicast and multicast traffic transport and get the same convergence time for unicast and multicast traffic. 2) multi-destination transport simplification: rely on the LSDB for computing a distribution tree and not run PIM hello protocol. 3) forwarding efficiency: no need to always forward the traffic to the RP; 4) better scalability: no need to maintain heavy PIM soft states. TRILL [RFC6325] has used IS-IS for both single destination and multi-destination packet transport, which proves the protocol capability of doing both.

1.1. Conventions used in this document

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 RFC-2119 [RFC2119].

2. IS-IS Protocol Extension

2.1. RTADDR sub-TLV

This is a sub-TLV that is used in either a Router Capabilities TLV or an MT Capabilities TLV. Each RTADDR sub-TLV contains a root IPv4 address and multicast group addresses that associate to the tree. A router may use multiple RTADDR sub-TLVs to announce multiple root addresses and associated multicast groups with each root. RTADDR sub-TLV format is below.

```

+-----+
|subType=RTADDR | (1 byte)
+-----+
| Length | (1 byte)
+-----+
| Root IPv4 Address |
+-----+
|S| RESV | (1 byte)
+-----+
| Tree Priority | (1 byte)
+-----+
|Num of Groups | (1 byte)
+-----+
| Group Address (1) |
+-----+
| Group Mask (1) |
+-----+
~
+-----+
| GROUP Address (N) |
+-----+
| Group Mask (N) |
+-----+

```

Where:

subType: RTADDR (TBD)

Length: variable depending on the number of associated groups

Root IPv4 Address: IPv4 Address for a root

S bit: If set, the rooted tree for single area only. Otherwise, the rooted tree crosses multiple areas.

RESV: 3 reserved bits. MUST be sent as zero and ignored on receipt.

Tree Priority: An eight bit unsigned integer where larger magnitude means higher priority. Zero means no priority.

Num of Groups: the number of group addresses

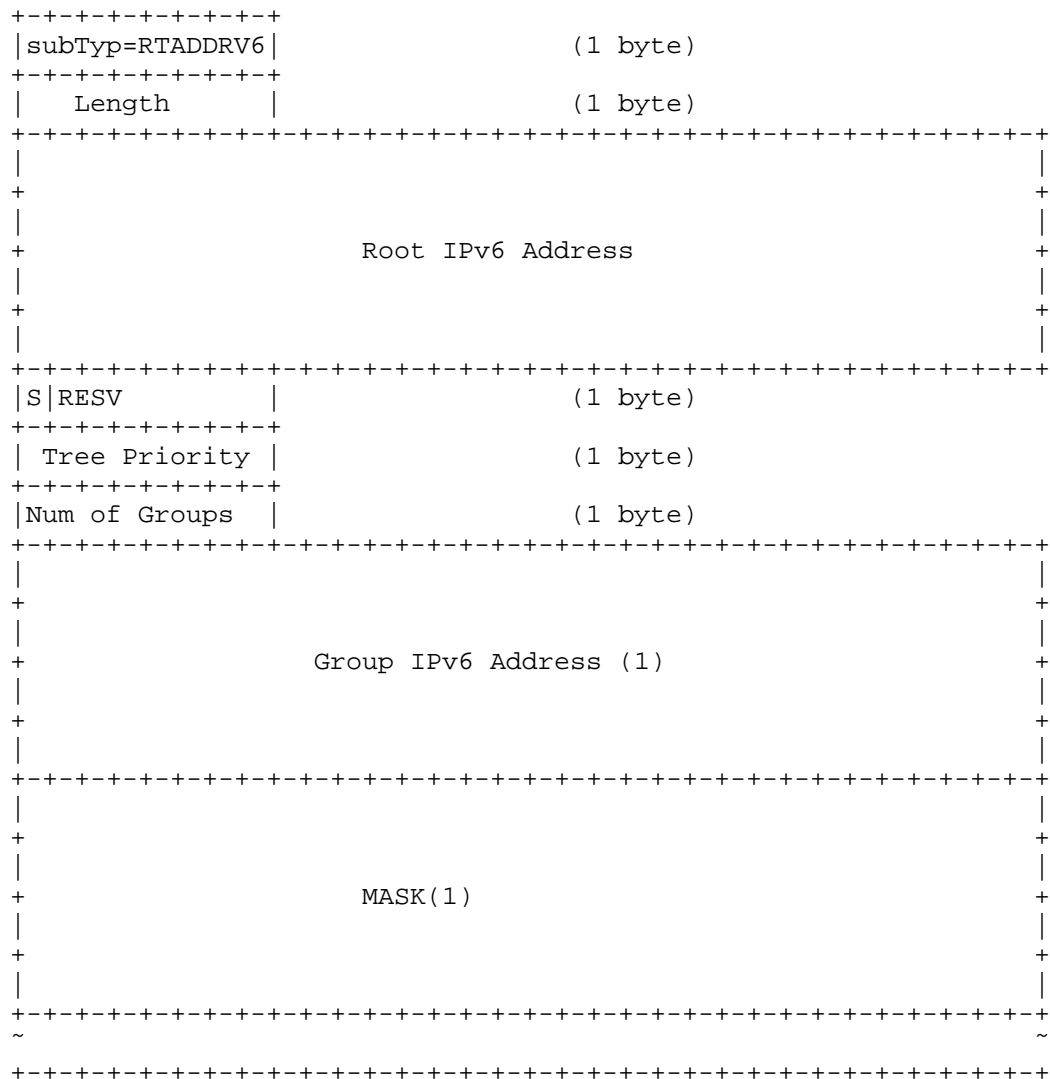
Group Address: IPv4 Address for the group

Group Mask: multicast group range

One router may be the root for multiple trees. Each tree associates to a set of multicast groups. In this case, a router encodes multiple RTADDR sub-TLVs to announce root addresses, one for each root, in either a Router Capabilities TLV or an MT Capabilities TLV. The group address/mask in different sub-TLVs can overlap. See section 3 for detail.

2.2. RTADDRV6 sub-TLV

This sub-TLV is used in an IPv6 network. It has the same format and usage except that the addresses are in IPv6.



2.3. The Group Address Sub-TLV

The Group Address TLV and a set of Group Address sub-TLVs are defined in RFC 7176 [RFC7176]. The GIP-ADDR and GIPV6-ADDR sub-TLVs are used in this solution. An edge router uses the GIP-ADDR sub-TLV or GIPV6-ADDR to announce its interested multicast groups. The GIP-

ADDR sub-TLV applies to an IPv4 network and GIPV6-ADDR sub-TLV for IPv6 network.

When using a GIP-ADDR or GIPV6-ADDR sub-TLV, the field VLAN-ID MUST set to zero and be ignored. Other field usage remains the same as [RFC7176]

3. Procedures

When an operator selects a router as a distribution tree root, he/she configures the tree root address and associated multicast groups on the router. A tree root address can be an interface address or router loopback address. After the configuration, the router will include a RTADDR sub-TLV, inside either a Router Capabilities TLV or an MT Capabilities TLV, where the tree root address and multicast groups are specified. If multiple trees are configured on the router, multiple RTADDR sub-TLVs are added in one or more Router Capabilities TLVs or MT Capabilities TLVs to specify individual tree roots. For IPv4 network, RTADDR sub-TLV is used. For IPv6, RTADDRV6 sub-TLV is used. Note that the rest of document specifies the processes for an IPv4 network only. The processes for an IPv6 network are the same.

Operators may associate one multicast group to more than one tree for the redundancy purposes and use the tree priority to specify the primary tree preference. Section 3.2 describes the primary tree selection.

3.1. Distribution Tree Computation

Upon receiving RTADDR sub-TLVs, routers track the tree roots and associated multicast groups. When the LSDB stabilizes, routers calculate all rooted trees according to the LSDB and shortest path algorithm.

One multicast group may associate to multiple trees. It is important that all the routers choose the same tree for a multicast group. Section 3.2 and 3.3 describes the tiebreaking rule for primary tree selection for a multicast group and parent selection in case of equal-cost to potential children.

3.2. Parent Selection

It is important, when building a distribution tree, that all routers choose the same links for the tree. Therefore, when there are equal costs from a potential child node to possible parent nodes, all

routers need to use the same tiebreakers. It is also desirable to allow splitting of traffic on as many links as possible in such situations. TRILL [RFC6325] achieves this by defining multiple rooted trees and using the tiebreakers to enable nodes in these trees to choose different parents. This draft uses the same tiebreakers as TRILL ([RFC6325] as clarified and updated by section 3.4 and 3.5 of [RFC7180]), and states as follow:

If there are k distribution trees in the network, when each router computes these trees, the k trees calculated are ordered and numbered from 0 to $k-1$ in ascending order according to root IP addresses.

The tiebreaker rule is: When building the tree number j , remember all possible equal cost parents for router N . After calculating the entire "tree" (actually, directed graph), for each router N , if N has " p " parents, then order the parents in ascending order according to the 7-octet IS-IS ID considered as an unsigned integer, and number them starting at zero. For tree j , choose N 's parent as choice $(j-1) \bmod p$.

3.3. Parallel Local Link Selection

If there are parallel point-to-point links between two routers, say $R1$ and $R2$, these parallel links would be visible to $R1$ and $R2$, but not to other routers. If this bundle of parallel links is included in a tree, it is important for $R1$ and $R2$ to decide which link to use; if the $R1$ - $R2$ link is the branch for multiple trees, it is desirable to split traffic over as many link as possible. However the local link selection for a tree is irrelevant to other Routers. Therefore, the tiebreaking algorithm need not be visible to any Routers other than $R1$ and $R2$.

When there are L parallel links between $R1$ and $R2$ and they both are on K trees. L links are ordered from 0 to $L-1$ in ascending order of Circuit ID as associated with the adjacency by the router with the highest System ID, and K trees are ordered from 0 to $K-1$ in ascending order of root IP addresses. The tiebreaker rule is: for tree k , select the link as choice $k \bmod L$.

Note that if multiple distribution trees are configured in a network or on a router, better load balance among parallel links through the tie-breaking algorithm can be achieved. Otherwise, if there is only one tree is configured, then only one link in parallel links can be used for the corresponding distribution tree. However, calculating and maintaining many trees is resource consuming. Operators need to balance between two.

3.4. Tree Selection for a Group

Routers receive one or more possible multicast group-range-to-tree mappings. Each mapping specifies a range of multicast groups. It is possible that a group-range is associated with multiple trees that may have the same or different priority. When a multicast group-range associates with more than one tree, all routers have to select the same tree for the group-range. The tiebreaker rules specified in PIM [RFC4601] are used. They are:

- o Perform longest match on group-range to get a list of trees.
- o Select the tree with highest priority.
- o If only one tree with the highest priority, select the tree for the group-range.
- o If multiple trees are with the highest priority, use the PIM hash function to choose one. PIM hash function is described in section 4.1.1 in RFC4601 [RFC4601].

3.5. Pruning a Distribution Tree for a Group

Routers prune the distribution tree for each associated multicast group, i.e. eliminating branches that have no potential downstream receivers. Multi-destination packets SHOULD only be forwarded on branches that are not pruned. The assumption here is that a multicast source is also a multicast receiver but a multicast receiver may not be a multicast source.

Routers prune the trees based on the groups specified in GRADD-TLV from edge routers. Routers maintain a list of adjacency interfaces that are on the pruned tree for a multicast group. Among these interfaces, one interface may be toward the tree-root router and other are toward the egress routers.

3.6. Reverse Path Forwarding Check (RPFC)

The routing transients resulting from topology changes can cause temporary transient loops in distribution trees. If no precautions are taken, and there are fork points in such loops, it is possible for multiple copies of a packet to be forwarded. If this is a problem for a particular use, a Reverse Path Forwarding Check (RPFC) may be implemented.

In this case, the RPFC works by a router determining for each port, based on the source and destination IP address of a packet, whether the port is a port that router expects to receive such a packet. In other words, is there an edge router with reachability to the source IP address such that, starting at that router and using the tree indicated by the destination IP address, the packet would have arrived at the port in question. If so, it is further distributed. If not, it is discarded. An RPFC can be implemented at some routers and not at others.

3.7. Forwarding Using a Pruned Distribution Tree

Forwarding a multi-destination packet follows the pruned tree for the group that the packet belongs to. It is done as follows.

- o If the router receives a multi-destination packet with group IP address that does not associated with any tree, the packet **MUST** be dropped.
- o Else check if the link that the packet arrives on is one of the ports in the pruned distribution tree. If not, the packet **MUST** be dropped.
- o Else perform RPF checking (section 3.5). If it fails, the packet **SHOULD** be dropped.
- o Else the packet is forwarded onto all the adjacency interfaces in the list for the group except the interface where the packet receive.

3.8. Local Forwarding at Edge Router

Upon receiving a multi-destination packet, besides forwarding it along the pruned tree, an edge router may also need to forward the packet to the local hosts attached to it. This is referred to as local forwarding in this document.

The local group database is needed to keep track of the group membership of the router's directly attached network or host. Each entry in the local group database is a [group, network/host] pair, which indicates that the attached network has one or more hosts belonging to the multicast group. When receiving a multi-destination packet, the edge router forwards the packet to the network/host that match the [group, network/host] pair in the local group database.

The local group database is built through the operation of the IGMPv3 [RFC3376]. When an edge router becomes Designated Router on

an attached network, say N1, it starts sending periodic IGMPv3 Host Membership Queries on the network. Hosts then respond with IGMPv3 Host Membership Reports, one for each multicast group to which they belong. Upon receiving a Host Membership Report for a multicast group A, the router updates its local group database by adding/refreshing the entry [Group A, N1]. If at a later time Reports for Group A cease to be heard on the network, the entry is then deleted from the local group database. The Designated further sends the LSP message with GRADDR sub-TLV to inform other routers about the group memberships in the local group database. A router MUST ignore Host Membership Reports received on those networks where the router has not been elected Designated Router.

When the solution described in this document applies to the underlying network that transports overlay virtual networks [NVO3FRWK], A Designated Router further necessarily maintains the mapping between an overlay multicast group and a underlying multicast group, and performs packet encapsulation/descapsulation upon receiving a packet from host or the underlying network. Mapping between an overlay multicast group and a underlying multicast group can be manually configured, automatically generated by an algorithm, or dynamically informed at a Designated Router. The same edge router should be selected as the Designated Router for the overlay multicast group and underlying multicast group that are associated. The mapping method is beyond the scope of this document.

3.9. Distribution Tree across different IGP Levels

An IGP (Interior Gateway Protocol) network may be designed as a multi-area network for the scalability, faster-convergence. Multicast sources and listeners may be in the same or different areas. The former is a special case of the latter. To support multi-destination transport over multi-areas, it is necessary to build a distribution tree across areas and prune the tree based on the listener locations, i.e. interested edge routers that may reside in different areas.

For an IS-IS multi-area network, there are level1 and level2 routers as well as level1/2 (border) routers. A level1 router only has the router/topology information for its area. A level2 router has router/topology information for level2 area as well as reachability information for level1 areas. A border router participates in both level1 and level2 areas and has the router/topology information for

level2 and all directly attached level1 areas but maintains separate LSDBs for level2 and each attached level 1 area. Traffic from one

area to another area must traverse through a border router. It is possible to have more than one border router between two areas for resilience.

To build a distribution tree across mutli-areas, an operator can select a tree-root node for a set of multicast groups. The node can be in levell1 area or level2 area. All the nodes including border nodes in the area compute the distribution tree as described in section 3.1-3.4. Border routers automatically select a designated forwarder for the multicast groups associated to the tree (see below). The border router selected as designate forwarder (DF) announces itself as the tree root in the adjacent area if the S bit in the RTADDR TLV is clear. The nodes in the adjacent area will compute the distribution tree in the same way. Note that a border router may be the tree-root in the adjacent area for the multicast groups that may associate with different trees. If S bit in the RTADDR TLV is set, the rooted distribution tree is only built in the area where the root node resides.

The document specifies following additional rules for a border router that supports the multicast mechanism described here. The rules apply to the case of the distribution tree across multiple areas.

If a border router is selected as designated forwarder in adjacent area for a set of multicast groups, it should perform following:

- o It MUST track the group-memberships in its participated areas.
- o It MUST send a summary group membership of one area to the adjacent area as of an edge router.
- o It performs the pruning process in each area, respectively, based on the received group-membership LSPs from that area.
- o When receiving multicast traffic from one area, it forwards the packet along the pruned tree into the adjacent area.
- o Optionally performs reverse path forwarding check (RPFC)

If a border router is not selected as the designated forwarder for the multicast groups, the followings apply:

- o It SHOULD NOT propagate the group-membership information of one area to any other areas. It SHOULD remove the TLV before forwarding it.

- o It SHOULD NOT forward multicast group traffic to another adjacent area. It SHOULD discard such traffic.

Selecting a border router as the designated forwarder of multicast group traffic may be done manually or automatically.

4. Mobility Support

4.1. Listener moves from one edge router to another

When listener moves from one edge router, say E1, to another, say E2. E1 will detect the host left and send IGMP query for (S, G). Upon the listener join E2, if E2 has not joined (S,G), E1 should announce itself as listener to the (S,G) tree.

4.2. Source host moves from one edge router to another

Multicast Tree reaches to every edge router, so source host mobility is supported naturally. If RPFC is used on a router, the port that router expects to receive packet may change. Thus, the notification on source host moves is necessary.

5. Backward Compatibility

If a router does not support the distribution tree function described in this document, distribution tree computation MUST NOT include this router. This may result the incomplete tree. An operator can build a tunnel between two routers, which allows a single rooted tree to be built. How to build the tunnel is outside scope of this document.

6. Interworking with PIM

It may be desirable for IS-IS multicast to interwork with PIM on the same network domain or different domains. The interworking solution is for further evaluation.

7. Security Considerations

For the further study.

8. IANA Considerations

IANA is requested to assign two new sub-TLV numbers for RTADDR and RTADDRV6 as specified in Sections 2.1 and 2.2. These sub-TLVs can be used under both the Router Capability (#242) and MT Capability (#144)

TLVs. To avoid confusion, each sub-TLV should be assigned the same sub-Type number under each of these two TLVs.

9. Acknowledgements

Authors like to thank Mike McBride and Linda Dunbar for their valuable inputs.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC2119, March 1997.
- [RFC3376] Cain B., etc, "Internet Group Management Protocol, Version 3", rfc4604, October 2002
- [RFC4601] Fenner, B., et al, "Protocol Independent multicast - Sparse Mode (PIM-SM): Protocol Specification", rfc4601, August 2006
- [RFC5015] Handley, M., et al, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", rfc5015, October 2007
- [RFC5120] Przygienda, T., et al, "M-ISIS: Multi Topology (MT) Routing in Intermediate System to Intermediate Systems (IS-ISs)", rfc5120, February 2008
- [RFC6325] Perlman, R., et al, "Routing Bridges (RBridges): Base Protocol Specification", RFC6325, July 2011
- [RFC7176] Eastlake 3rd, D., Senevirathne, T., Ghanwani, A., Dutt, D., and A. Banerjee, "Transparent Interconnection of Lots of Links (TRILL) Use of IS-IS", RFC 7176, May 2014.

10.2. Informative References

- [DCMCAST] McBride, M., Lui, H., "Multicast in the Data Center Overview", draft-mcbride-armd-mcast-overview, 2012
- [MCASTISS] Ghanvani, A., "Multicast Issues in Networks Using NVO3", draft-ghanwani-nvo3-mcast-issues, work in progress

[NVO3FRWK] Lasserre, M., "Framework for DC Network Virtualization",
draft-ietf-nvo3-framework, work in progress.

[VXLAN] Mahalingam, M., Dutt, D., etc, "VXLAN: A Framework for
Overlaying Virtualized Layer 2 Networks over Layer 3
Networks", draft-mahalingam-dutt-dcops-vxlan, work in
progress

Authors' Addresses

Lucy Yong
Huawei USA
5340 Legacy Drive
Plano, TX 75025 USA

Phone: 469-277-5837
Email: lucy.yong@huawei.com

Weiguo Hao
Huawei Technologies
101 Software Avenue,
Nanjing 210012
China

Phone: +86-25-56623144
Email: haoweiguo@huawei.com

Donald Eastlake
Huawei
155 Beaver Street
Milford, MA 01757 USA

Phone: +1-508-333-2270
EMail: d3e3e3@gmail.com

Andrew Qu
MediaTek
San Jose, CA 95134 USA

Email: laodulaodu@gmail.com

Jon Hudson
Brocade
130 Holger Way

San Jose, CA 95134 USA

Phone: +1-408-333-4062

Email: jon.hudson@gmail.com

