

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 19, 2014

S. Litkowski
Orange
June 17, 2014

Yang Data Model for ISIS protocol
draft-litkowski-netmod-isis-cfg-00

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 19, 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

Internet-Draft

isis-cfg

June 2014

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
2.	Design of the data model	2
2.1.	Authentication parameters	4
2.2.	Multitopology parameters	4
2.3.	Per level parameters	4
2.4.	Per interface parameters	5
2.5.	Operational states	6
3.	Yang module	6
4.	Security Considerations	22
5.	Acknowledgements	22
6.	IANA Considerations	22
7.	Normative References	22
	Author's Address	22

[1.](#) Introduction

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

[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-op : that contains read-only states.

Each of this container contains a list of instances as many implementations are currently supporting multiple ISIS instances within a single virtual-router.

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

```
module: isis
  +--rw isis-cfg
  |
  +--rw instances
```

```

|      +--rw instance [name]
|      |      +--rw name
|      |      +--rw enabled
|      |      +--rw isis-level
|      |      +--rw area

```

```

|      +--rw system-id
|      +--rw ipv4-router-id
|      +--rw ipv6-router-id
|      +--rw reference-bandwidth
|      +--rw lsp-mtu
|      +--rw lsp-lifetime
|      +--rw lsp-refresh
|      +--rw isis-authentication-cfg
|      |      ...
|      +--rw multi-topology-cfg
|      |      ...
|      +--rw isis-level-1-cfg
|      |      ...
|      +--rw isis-level-2-cfg
|      |      ...
|      +--rw overload
|      |      ...
|      +--rw interfaces
|      |      +--rw interface [name]
|      |      |      ...
|
+--ro isis-op
  +--ro instances
    +--ro instance [name]
      +--ro adjacencies
        |      +--ro adjacency [interface]
        |      |      +--ro interface
        |      |      +--ro level
        |      |      +--ro state
        +--ro spf-log
          |      +--ro event [id]
          |      |      +--ro id
          |      |      +--ro level
          |      |      +--ro spf-type
          |      |      +--ro spf-delay
          |      |      +--ro schedule-timestamp

```

```

|      +--ro start-timestamp
|      +--ro end-timestamp
|      +--ro trigger-lsp [lsp]
|          +--ro lsp
|          +--ro sequence
+--ro lsp-log
|   +--ro event [id]
|       +--ro id
|       +--ro level
|       +--ro lsp
|           +--ro lsp
|           +--ro sequence

```

```

|      +--ro received-timestamp
+--ro database
|   +--ro level-1
|       ...
|   +--ro level-2
|       ...
+--ro hostnames
|   +--ro hostname [system-id]
|       +--ro system-id
|       +--ro hostname

```

[2.1.](#) Authentication parameters

The following figure describes the global authentication parameters for a particular ISIS instance.

```

+--rw isis-authentication-cfg
|   +--rw psnp-authentication
|   +--rw csnp-authentication
|   +--rw hello-authentication
|   +--rw authentication-key
|   +--rw authentication-type

```

[2.2.](#) Multitopology parameters

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

```
+--rw multi-topology-cfg
  +--rw ipv4-unicast
  +--rw ipv6-unicast
  +--rw ipv4-multicast
  +--rw ipv6-multicast
```

[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 ...

```
+--rw isis-level-1-cfg
  +--rw enabled
  +--rw psnp-authentication
  +--rw csnp-authentication
  +--rw hello-authentication
  +--rw authentication-key
  +--rw authentication-type
  +--rw metric-type
  +--rw preference
  +--rw external-preference
  +--rw default-ipv4-unicast-metric
  +--rw default-ipv6-unicast-metric
  +--rw default-ipv4-multicast-metric
  +--rw default-ipv6-multicast-metric
```

[2.4.](#) Per interface parameters

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

```
+--rw interface [name]
```

```

+--rw name
+--rw level
+--rw lsp-interval
+--rw passive
+--rw csnp-interval
+--rw hello-authentication-type
+--rw hello-authentication-key
+--rw hello-interval
+--rw hello-multiplier
+--rw hello-padding
+--rw ipv4-unicast
+--rw ipv6-unicast
+--rw ipv4-multicast
+--rw ipv6-multicast
+--rw interface-type
+--rw enabled
+--rw tag
+--rw level-1
|   +--rw hello-authentication-type
|   +--rw hello-authentication-key
|   +--rw hello-interval
|   +--rw hello-multiplier
|   +--rw ipv4-unicast
|   +--rw ipv6-unicast
|   +--rw ipv4-multicast
|   +--rw ipv6-multicast

```

```

|   +--rw priority
|   +--rw ipv4-unicast-metric
|   +--rw ipv6-unicast-metric
|   +--rw ipv4-multicast-metric
|   +--rw ipv6-multicast-metric
|   +--rw passive
|
+--rw level-2
    +--rw hello-authentication-type
    +--rw hello-authentication-key
    +--rw hello-interval
    +--rw hello-multiplier
    +--rw ipv4-unicast
    +--rw ipv6-unicast
    +--rw ipv4-multicast

```

```

+--rw ipv6-multicast
+--rw priority
+--rw ipv4-unicast-metric
+--rw ipv6-unicast-metric
+--rw ipv4-multicast-metric
+--rw ipv6-multicast-metric
+--rw passive

```

[2.5.](#) Operational states

isis-op 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.](#) Yang module

```

module isis {
  namespace TBD;

  prefix isis;

```

```

import ietf-routing {
  prefix "rt";
}

import ietf-inet-types {
  prefix inet;
}

```

```

import ietf-yang-types {
    prefix yang;
}

contact
    "Stephane Litkowski

    Email : stephane.litkowski@orange.com";

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

revision 2014-06-11 {
    description "Initial revision.";
}
identity isis {
    base rt:routing-protocol;
    description "Identity for the ISIS routing protocol.";
}
typedef isis-level {
    type enumeration {
        enum level-1;
        enum level-2;
        enum level-1-2;
    }
}

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

typedef isis-std-metric {
    type uint8 {
        range "0 .. 63";
    }
}

```

```

grouping isis-route-content {

```



```

    leaf metric {
        type uint32;
    }
    leaf-list tag {
        type uint32;
    }
    leaf route-type {
        type enumeration {
            enum l2-up-internal;
            enum l1-up-internal;
            enum l2-up-external;
            enum l1-up-external;
            enum l2-down-internal;
            enum l1-down-internal;
            enum l2-down-external;
            enum l1-down-external;
        }
    }
}

augment "/rt:routing/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;
}

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

grouping isis-prefix-ipv4-std {
    leaf up-down {
        description "This leaf expresses the value of up/down bit.";
        type boolean;
    }
    leaf i-e {
        description "This leaf expresses the value of I/E bit.";
        type boolean;
    }
    leaf ip-prefix {
        type inet:ipv4-address;
    }
    leaf prefix-len {

```

```
        type uint8;
    }
    leaf default-metric {
        type isis-std-metric;
    }
    leaf delay-metric {
        type isis-std-metric;
    }
    leaf expense-metric {
        type isis-std-metric;
    }
    leaf error-metric {
        type isis-std-metric;
    }
}

grouping isis-prefix-ipv4-extended {
    leaf up-down {
        description "This leaf expresses the value of up/down bit.";
        type boolean;
    }
    leaf ip-prefix {
        type inet:ipv4-address;
    }
    leaf prefix-len {
        type uint8;
    }

    leaf metric {
        type isis-wide-metric;
    }
    leaf-list SUBTLV1 {
        type uint32;
    }
}

grouping isis-prefix-ipv6-extended {
    leaf up-down {
        description "This leaf expresses the value of up/down bit.";
        type boolean;
    }
    leaf ip-prefix {
        type inet:ipv6-address;
    }
    leaf prefix-len {
        type uint8;
    }
}
```

}

Internet-Draft

isis-cfg

June 2014

```
    leaf metric {
        type isis-wide-metric;
    }
    leaf-list SUBTLV1 {
        type uint32;
    }
}

grouping isis-neighbor-extended {
    leaf system-id {
        type string;
    }
    leaf metric {
        type isis-wide-metric;
    }
}

grouping isis-database {
    leaf id {
        type string;
    }
    leaf checksum {
        type uint16;
    }
    leaf sequence {
        type uint32;
    }
    leaf attributes {
        type bits {
            bit PARTITIONED;
            bit ATTACHED-ERROR;
            bit ATTACHED-EXPENSE;
            bit ATTACHED-DELAY;
            bit ATTACHED-DEFAULT;
            bit OVERLOAD;
        }
    }
    leaf isis-level {
        type isis-level;
    }
}
```

```

}

container TLV10 {
    description "This leaf describes authentication information of the
leaf authentication-type {
        type uint8;
    }
    leaf authentication-key {
        type string;
    }
}

```

```

    }
}

container TLV22 {
    list neighbor {
        key "system-id";
        uses isis-neighbor-extended;
    }
}

container TLV128 {
    list ip-internal-reachability {
        key "ip-prefix";
        uses isis-prefix-ipv4-std;
    }
}

leaf-list TLV129 {
    description "This leaf describes the protocol supported by the node";
    type uint8;
}

container TLV130 {
    list ip-external-reachability {
        key "ip-prefix";
        uses isis-prefix-ipv4-std;
    }
}

leaf-list TLV132 {
    description "This leaf describes the ipv4 addresses of the node.";
}

```

```

        type inet:ipv4-address;
    }

    leaf TLV134 {
        description "This leaf describes the IPv4 Traffic Engineering route"
        type inet:ipv4-address;
    }

    container TLV135 {
        list extended-ip-reachability {
            key "ip-prefix";
            uses isis-prefix-ipv4-extended;
        }
    }

    leaf TLV137 {

```

Litkowski

Expires December 19, 2014

[Page 11]

Internet-Draft

isis-cfg

June 2014

```

        description "This leaf describes the name of the node.";
        type string;
    }

    leaf TLV140 {
        description "This leaf describes the IPv6 Traffic Engineering route"
        type inet:ipv6-address;
    }

    container TLV222 {
        list neighbor {
            key "system-id";
            leaf MT-ID {
                type uint16 {
                    range "0 .. 4095";
                }
            }
            uses isis-neighbor-extended;
        }
    }

    container TLV229 {

```

```

description "This leaf describes the MT entries.";
list topology {
    key "MT-ID";

    leaf MT-ID {
        type uint16 {
            range "0 .. 4095";
        }
    }

    leaf attributes {
        type bits {
            bit OVERLOAD;
            bit ATTACHED;
        }
    }
}

leaf-list TLV232 {
    description "This leaf describes the ipv6 addresses of the node.";
    type inet:ipv6-address;
}

```

```

container TLV235 {
    list extended-ip-reachability {
        key "ip-prefix";
        leaf MT-ID {
            type uint16 {
                range "0 .. 4095";
            }
        }
        uses isis-prefix-ipv4-extended;
    }
}

container TLV237 {
    list extended-ip-reachability {
        key "ip-prefix";
        leaf MT-ID {
            type uint16 {

```

```

        range "0 .. 4095";
    }
}
uses isis-prefix-ipv6-extended;

}

}

container TLV236 {
    list extended-ipv6-reachability {
        key "ip-prefix";
        uses isis-prefix-ipv6-extended;
    }
}

container TLV242 {
    description "This leaf describes the capabilities of the node. This
    leaf binary {
        type binary;
    }
}

}

grouping address-family-cfg {
    leaf ipv4-unicast {
        description "This leaf defines if IPv4 unicast is activated.";
        type boolean;
    }
    leaf ipv6-unicast {
        description "This leaf defines if IPv6 unicast is activated.";

```

```

        type boolean;
    }
    leaf ipv4-multicast {
        description "This leaf defines if IPv4 multicast is activated.";
        type boolean;
    }
    leaf ipv6-multicast {
        description "This leaf defines if IPv6 multicast is activated.";
        type boolean;
    }
}

```

```

grouping interface-hello-cfg {
    leaf hello-authentication-type {
        type enumeration {
            enum    none;
            enum    plaintext;
            enum    message-digest;
        }
    }
    leaf hello-authentication-key {
        type string;
    }
    leaf hello-interval {
        description "This leaf defines the interval of hello messages.";
        type uint16;
    }
    leaf hello-multiplier {
        description "This leaf defines the number of hello failed to be received";
        type uint16;
    }
}

grouping interface-level-cfg {
    uses interface-hello-cfg;

    uses address-family-cfg;

    leaf priority {
        type uint8 {
            range "0 .. 127";
        }
    }
    leaf ipv4-unicast-metric {
        type isis-wide-metric;
    }
    leaf ipv6-unicast-metric {
        type isis-wide-metric;
    }
}

```

```

}
leaf ipv4-multicast-metric {
    type isis-wide-metric;
}

```



```

    leaf ipv6-multicast-metric {
        type isis-wide-metric;
    }
    leaf passive {
        description "This leaf defines if interface is in passive mode (ISIS)"
        type boolean;
        default "false";
    }
}

grouping interface-cfg {
    description "ISIS is enabled on interfaces that have an entry in this l
    leaf name {
        type leafref {
            path "/rt:routing/rt:router/rt:interfaces/rt:interfaces/rt:inte
        }
    }
    leaf level {
        description "This leaf defines the associated ISIS level of the int
        type isis-level;
    }
    leaf lsp-interval {
        description "This leaf defines the interval between LSP transmissio
        type uint16;
    }
    leaf passive {
        description "This leaf defines if interface is in passive mode (ISI
        type boolean;
        default "false";
    }
    leaf csnp-interval {
        type uint16;
    }
}

uses interface-hello-cfg;

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

uses address-family-cfg;

```

```
    leaf interface-type {
        type enumeration {
            enum broadcast;
            enum point-to-point;
        }
    }

    leaf enabled {
        type boolean;
        default "true";
    }

    leaf-list tag {
        description "This leaf defines list of tags associated with the int
        type uint32;
    }

    container level-1 {
        uses interface-level-cfg;
    }

    container level-2 {
        uses interface-level-cfg;
    }
}

grouping isis-authentication-cfg {
    leaf psnp-authentication {
        type boolean;
        default "true";
    }
    leaf csnp-authentication {
        type boolean;
        default "true";
    }
    leaf hello-authentication {
        type boolean;
        default "true";
    }
    leaf authentication-key {
        type string;
    }
    leaf authentication-type {
        type enumeration {
            enum none;
            enum plaintext;
            enum message-digest;
```

```
}
```

Internet-Draft

isis-cfg

June 2014

```
    }  
}
```

```
grouping isis-level-cfg {
```

```
    leaf enabled {  
        description "This leaf defines the status of the administrative sta  
        type boolean;  
        default "true";  
    }
```

```
    uses isis-authentication-cfg;
```

```
    leaf metric-type {  
        type enumeration {  
            enum wide-only;  
            enum old-only;  
            enum both;  
        }  
    }
```

```
    leaf preference {  
        description "This leaf defines the protocol preference.";  
        type uint8;  
    }
```

```
    leaf external-preference {  
        description "This leaf defines the protocol preference for external  
        type uint8;  
    }
```

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

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

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

```

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

augment "/rt:routing/rt:routing-instance/rt:routing-protocols/rt:routing-pr
    when "rt:type = 'isis:isis'" {

```

```

    description "This augment is only valid when routing protocol insta
}
container isis-cfg {
    container instances {
        description ".";
        list instance {
            key "name";
            description "Defines the list of ISIS instances.";
            leaf name {
                type string;
            }
            leaf enabled {
                type boolean;
                default "true";
            }
            leaf isis-level {
                type isis-level;
            }
            leaf area-id {
                type string;
            }

            leaf system-id {
                type string;
            }
            leaf ipv4-router-id {
                description "Router ID value that would be used in TLV1
                type inet:ipv4-address;
            }
            leaf ipv6-router-id {
                description "Router ID value that would be used in TLV2
                type inet:ipv6-address;
            }
        }
    }
}

```

```

leaf reference-bandwidth {
    description "This leaf defines the bandwidth for calcul
    type uint32;
}

leaf lsp-mtu {
    description "This leaf describes the maximum size of a
    type uint16;
}
leaf lsp-lifetime {
    description "This leaf describes the lifetime of the ro
    type uint16;
}
leaf lsp-refresh {
    description "This leaf describes the refresh interval o

```

Litkowski

Expires December 19, 2014

[Page 18]

Internet-Draft

isis-cfg

June 2014

```

    type uint16;
}

uses isis-authentication-cfg;

container multi-topology-cfg {
    description "This container describes activation of MT
    uses address-family-cfg;
}

container isis-level-1-cfg {
    description "Defines configuration parameters of level

    uses isis-level-cfg;
}

container isis-level-2-cfg {
    description "Defines configuration parameters of level

    uses isis-level-cfg;
}

container overload {
    description "This leaf describes if the router is set t
    leaf status {
        type boolean;

```

```

        }
        leaf timeout {
            type uint16;
        }
    }

    container interfaces {
        list interface {
            key "name";
            uses interface-cfg;
        }
    }
}

container isis-op {
    config false;
    container adjacencies {
        description "This container lists the adjacencies of the local"
        list adjacency {
            key interface;

```

```

        leaf interface {
            type string;
        }
        leaf level {
            type isis-level;
        }
        leaf state {
            type enumeration {
                enum "Up";
                enum "Down";
                enum "Init";
            }
        }
    }
}

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

```

```

    leaf id {
        type uint32;
    }
    leaf spf-type {
        type enumeration {
            enum full;
            enum incremental;
            enum route-only;
        }
    }
    leaf level {
        type isis-level;
    }
    leaf spf-delay {
        description "This leaf describes the SPF delay that was";
        type uint32;
    }
    leaf schedule-timestamp {
        type yang:timestamp;
    }
    leaf start-timestamp {
        type yang:timestamp;
    }
    leaf end-timestamp {
        type yang:timestamp;
    }
    list trigger-lsp {
        key "lsp";

```

```

        leaf lsp {
            type string;
        }
        leaf sequence {
            type uint32;
        }
    }

}

}
container lsp-log {
    description "This container lists the LSP reception events.";

```

```

    list event {
        key id;

        leaf id {
            type uint32;
        }
        leaf level {
            type isis-level;
        }
        container lsp {
            leaf lsp {
                type string;
            }
            leaf sequence {
                type uint32;
            }
        }

        leaf received-timestamp {
            type yang:timestamp;
        }
    }
}
container database {
    container level-1 {
        list lsp {
            key id;

            uses isis-database;

        }
    }
    container level-2 {
        list lsp {
            key id;

```

```

        uses isis-database;

    }
}
}

```



```

        container hostnames {
            list hostname {
                key system-id;
                leaf system-id {
                    type string;
                }
                leaf hostname {
                    type string;
                }
            }
        }
    }
}

```

[4.](#) Security Considerations

[5.](#) Acknowledgements

[6.](#) IANA Considerations

[7.](#) 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.

Author's Address

Stephane Litkowski
 Orange

Email: stephane.litkowski@orange.com